Illia Danilishyn, Oleksandr Danilishyn

MATHEMATICAL FUNDAMENTALS OF INTERPRETATIONS THEORY (TYPES OF FUTURE SCIENCE) AND FUNDAMENTALLY NEW APPROACH TO ELEMENTARY PARTICLES, PHYSICS, BIOLOGY AND DYNAMIC PROGRAMMING BY PSELF-TYPE AND PASELF-TYPE

Monograph

Primedia eLaunch

Primedia eLaunch

Illia Danilishyn, Oleksandr Danilishyn

MATHEMATICAL FUNDAMENTALS OF INTERPRETATIONS THEORY (TYPES OF FUTURE SCIENCE) AND FUNDAMENTALLY NEW APPROACH TO ELEMENTARY PARTICLES, PHYSICS, BIOLOGY AND DYNAMIC PROGRAMMING BY PSELF-TYPE AND PASELF-TYPE

Monograph

Edited by *Volodymyr Pasynkov*

Shawnee, USA
Primedia eLaunch LLC
2026

Epigraph 1:
Mathematics is on the border of science (knowledge), so it is she who is able to make major breakthroughs beyond this border.

Epigraph 2:
The paradox is the truth

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 Interpretations Theory (Types of Future Science) and Fundamentally New Approach to Elementary Particles, Physics, Biology and Dynamic Programming by Pself-type and Paself-type: monograph / ed. by Pasynkov V. – Shawnee, USA : Primedia eLaunch LLC. – 2026. – 336 p.

ISBN 979-8-89898-334-5
DOI 10.36074/mfit.monograph-2026

Here are given mathematical fundamentals of interpretations theory (future science) and fundamentally new approach to elementary particles, induction, energy conservation, Dynamic Programming by pself-type and paself-type. Digitization of interpretations forms allows the use of digital technologies through appropriate programming. In science, there are two approaches to studying nature beyond classical science (conditionally, the (1, 1)-interpretation): the approach through quantum mechanics (conditionally, the (1, 2)-interpretation) and dynamic mathematics (conditionally, the (2, 1)-interpretation). Both approaches have the same "root": |||, but at two opposite ends: quantum mechanics ((1, 2)-interpretation) by $|||^{-1}$ by and dynamic mathematics (2, 1)-interpretation) by ||| and ((1, 2)-interpretation) by $|||^{-1}$ and other interpretations with using hierarchical structures of measures, some equations. Therefore, the conclusions are similar, but for different objects and processes. This article applies to physics, Dynamic Programming, and neural networks of the new direct-parallel and direct-accumulative types. For now, this is only an introductory article. The monograph first 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. For this, some aspects are proposed for consideration of Dynamic Science. $self_{science}^{our\ theory}$ - science here acts as a space for the application of our theory in the self-format, i.e., any place of science, in particular physics, can act as a place for the "location" of the self. It contains itself (accommodates any action C) in any place of science. On the basis of mathematical uncertainties, new mathematical structures are formed, allowing us to describe processes and objects that are fundamentally not determined by conventional deterministic methods. Objective uncertainties in any case can mean manifestations of processes and objects that are fundamentally not determined by conventional deterministic methods. Since dynamic mathematics places the primary emphasis on the dynamics of energy, rather than on objectivity, the level of approach to studying processes expands.Many energies are indeterminate because they are based on uncertainties from the perspective of traditional science—large concentrations of specific energy in a chaotic state. The foundation of dynamic mathematics lies in working with uncertainties, which makes it possible to manipulate these indeterminate energies using direct-accumulative direct-parallel neural networks. The second task of the monograph is to construct a new mathematical apparatus for neural networks of a fundamentally new type: direct-parallel and direct-accumulative action. We construct models of singularities for singular work with them through neural networks - analogues of the human CNS. Ordinary regular work with them in ordinary science is fundamentally unable to realize their capabilities. Therefore, singular science realized on a neural network - an analogue of the human CNS - will be much more natural. Unfortunately, we do not have funding to perform the necessary experiments and the practical creation of a technical model of such a neural network. 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. Social justice is fundamentally impossible as long as education (training) is based on achieving results, and not on the process. It is time for physicists to begin studying not only the manifestations of living energies, but also the living energies themselves, which are by no means expressed through objectivity and ordinary energies, although they are capable of manifesting themselves through a lower level - objectivity and ordinary energies. We, as mathematicians, offer a new corresponding apparatus for understanding nature and studying living energies. Significance of the article: 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. The significance of our monograph 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. 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.

UDC 004:51(07)

ISBN 979-8-89898-334-5

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 [1-5] in more interesting dynamic structures. Here is considered mathematical theory of interpretations, new paradoxical singularities (singularities of disintegration&synthesis), self-type singularities, analogues of equations for singularities,new approachis to elementary particles and physics, biology, Dynamic programming by pself-type and paself-type, which works through levels hierarchy in the space of energies with pseudo-living energies. 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' approachis 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 [20], 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.

Ukraine, December 2025
Volodymyr Pasynkov

Introduction

General interpretation form is H, i.e., interpretation is carried out through H. The next form is (A, B)-type, i.e., interpretation is carried out through (A, B).

D by form (A, (B, C)): here "cloth" $B\mid_C$ is for $D\mid_A$, i.e., $D\mid_{(B,C)}$, A, B, C, D – any. Here $B\mid_C$ is the interpretation for $D\mid_A$. For example, 1) D by form (2, ((Q, R), 2)) if A=1, B=2, C=1 we obtain self(D). "Cloth" search for natural and artificial phenomena and objects is science. Direct knowledge is ("Cloth" search for natural and artificial phenomena) ||| [19 - 21] ("Cloth" for natural and artificial phenomena).

We introduce the following preliminary hierarchies:

$$\left(\begin{array}{c}\text{Science by measure of self} - \text{type}\\ \text{Science by } \textit{manifestation} \text{ measure } \textit{of} \text{ self} - \text{type}\\ \textit{dynamic} \text{ Science}\\ \textit{usual} \text{ Science}\end{array}\right)$$

$$\left(\begin{array}{c}\text{Math by measure of self} - \text{type}\\ \text{Math by } \textit{manifestation} \text{ measure } \textit{of} \text{ self} - \text{type}\\ \textit{dynamic} \text{ Math}\\ \textit{usual} \text{ Math}\end{array}\right)$$

$$\left(\begin{array}{c}\text{any C by measure of self} - \text{type}\\ \text{any C by } \textit{manifestation} \text{ measure } \textit{of} \text{ self} - \text{type}\\ \textit{dynamic} \text{ any C}\\ \textit{usual} \text{ any C}\end{array}\right)$$

In fact, we give a static interpretation of Dynamic Mathematics, the real Dynamic Mathematics will be on SmnSprt. Dynamic Mathematics for Limbo between parallel lines. Any human action is carried out through Will, just a person learns to act in physical space through Will in accordance with the limitations of physical space, and in the energy space the magician learns through Will in accordance with the possibilities of energy space. The mind (the cerebral cortex) is responsible for interpretation in the form of physical space. The hypothalamus (subcortex of the brain) is responsible for interpretation in the form of energy space (direct knowledge). The position of the assemblage point on the surface of the human energy cocoon is responsible for interpretation in the form of physical space. The position of the assemblage point inside the human energy cocoon is responsible for interpretation in the form of energy space (direct knowledge). Usual attention is superficial perception. Ordinary science is "superficial" interpretation. Automorphism B is a relation by elements in B. self(B) is a holistic relation, where, for example, B contains itself as an element: $B \in B$. Conventional science is based on an element-by-element approach, our dynamic science is based on a

holistic approach and thus complements traditional science, since, for example, a holistic approach is needed to study living organisms. Thus, we come to holistic mathematics - Dynamic mathematics and we need devices and instruments with a holistic approach - real artificial intelligence. Dynamic Mathematics oriented toward the uniqueness of objects, elements, operations, and actions. Any interpretation is like “clothing” that we try on for nature, and our task is to find the most suitable “clothing” for nature. In Part I are given mathematical fundamentals of interpretations theory (future science). Other parts of the book apply Part I to Dynamic Mathematics, Dynamic Programming by pself-type and paself-type, and neural networks of the new direct-parallel and direct-accumulative types. Digitization of interpretations forms allows the use of digital technologies through appropriate programming. In science, there are two approaches to studying nature beyond classical science (conditionally, the (1, 1)-interpretation): the approach through quantum mechanics (conditionally, the (1, 2)-interpretation) and dynamic mathematics (conditionally, the (2, 1)-interpretation). Both approaches have the same "root": |||, but at two opposite ends: quantum mechanics ((1, 2)-interpretation) by $|||^{-1}$ [19 - 21] with $|||^{-1}$-characteristic, which is probability, and dynamic mathematics (2, 1)-interpretation) by ||| and ((1, 2)-interpretation) by $|||^{-1}$ and other interpretations with using hierarchical structures of measures. Therefore, the conclusions are similar, but for different objects and processes. For example,

the vector of energy levels of a non-living object:

$$\left(\begin{array}{c} \overset{\cdots}{} \\ parelf\,A\left(decignation-\overset{\equiv\equiv\equiv}{A}\right) \\ singelf\,A\left(decignation-\overset{\equiv\equiv}{A}\right) \\ subtle\ energy\ of\ object\ \ A\ \text{paradoxical}\ upper\ level\ (\ pa|||)\left(decignation-\overset{\equiv\equiv}{A}\right) \\ subtle\ energy\ of\ object\ \ \ A\ \text{paradoxical}\ mid-level(paself(A))\left(decignation-\overset{\equiv\equiv\equiv}{A}\right) \\ subtle\ energy\ of\ object\ \ \ A\ \text{paradoxical}\ down-level(pself(A))\left(decignation-\overline{\overline{\overline{A}}}\right) \\ \begin{array}{cc} / & \backslash \\ subtle\ energy\ of\ |||^{-1} & subtle\ energy\ of\ ||| \\ oself(A) & \widehat{A} \\ & \left(\overline{A}\right) \end{array} \\ ordinary\ energy\ exhibited\ by\ an\ \ object\ \ A\left(decignation-\underline{A}\right)\ \leftarrow\ \ the\ raw\ energy\ of\ an\ object\ \ A\left(decignation-A\right) \end{array}\right),$$

$\overline{A} = self(A)$, $\widehat{A} = self^{\frac{3}{2}}(containment\ for\ creating\ of\ A)$. Using hierarchical structures of measures, we will consider further in this part. At other levels of the hierarchy, unlike our usual level, we can consider only their elements—singularities, i.e., elements distinct from our usual level—and manifest them

through corresponding measures, relations, characteristics, and equations. At other levels of the hierarchy, unlike our usual level, there is neither our usual time nor our usual physical space. By changing the singularity at the upper level or introducing a new singularity by Dynamic science, Dynamic Programming by pself-type and paself-type with devices of a new type based on SmnSprt, it is possible to manifest them at the lower level, i.e., the usual one, and obtain what is needed.

Part I. SIIprt – 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 selfII-sets, selfII-elements, which is exactly what we need for new mathematical models for describing complex processes. Instead of the axiom of regularity, we introduce the other axioms [13].

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. Significance of this part: 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

$$\begin{matrix} C & A \\ DSIIprt & B \\ F & Q \end{matrix} \quad (*_{1.1}),$$

where A fits into B, B fits into Q, F is forced out from D, D is forced out from C simultaneously. A, B, D, C, F, Q are any, in particular, structures. The result of this process will be described by the expression

$$\begin{matrix} C & A \\ DSIIrt & B \\ F & Q \end{matrix} \quad (*_{1.2})$$

If A, B, D, C, F, Q are taken as sets, then we will call $(*_{1.1})$ a dynamic set.

It can be considered a simpler version of the dynamic set

$$\begin{matrix} & A & \\ SIIprt & B & (**_{1.1}), \\ & Q & \end{matrix}$$

where set A fits into set B, B fits into Q simultaneously, the result of this process will be described by the expression

$$\begin{matrix} & A & \\ SIIrt & B & (**_{1.2}) \\ & Q & \end{matrix}$$

or

$$\begin{matrix} C \\ BSIIprt(***_{1.1}) \\ A \end{matrix}$$

where set A is forced out from B, B is forced out from C, the result of this process will be described by the expression

$$\begin{matrix} C \\ BSIIrt(***_{1.2}) \\ A \end{matrix}$$

We consider the measure: $m^{**}(\begin{matrix} b & & A \\ D & SIIprt & R \\ F & & b \end{matrix}) = \frac{\mu(A)\mu(R)}{\mu(D)\mu(F)}$, where m(A),m(D), $\mu(R)$, $\mu(F)$ –usual measures of sets A, D, R, F respectively.

Remark. One can consider some generalization for $(*_{1.1})$: $\begin{matrix} q_1(C) & & A \\ q_4(D) & SIIprt & q(B) \\ F & & q_3(Q) \end{matrix}$, where A fits into B through q, q(B) fits into Q through q_3, F is forced out from D through

q_4, q_4(D) is forced out from C through q_1 simultaneously. The result of this process will be described by the expression $\begin{matrix} q_1(C) & A \\ q_4(D) & SIIrt \\ F & q_3(Q) \end{matrix} q(B)$.

Similarly, for $(**_{1.1})$: $\begin{matrix} A \\ SIIprt \\ q_3(Q) \end{matrix} q(B)$, where fits into B through q, q(B) fits into Q through q_3 F is forced out from D through q_4, q_4(D) is forced out from C through q_1 simultaneously, for $(***_{1.1})$: $\begin{matrix} q_1(C) \\ q_4(D) \\ F \end{matrix} SIIprt$, where F is forced out from D through q_4, q_4(D) is forced out from C through q_1 simultaneously. The result of this process will be described by the expression $\begin{matrix} q_1(C) \\ q_4(D) \\ F \end{matrix} SIIrt$.

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.

Remark. Similarly, all sections of this part can be generalized to a more general case:

$$\begin{matrix} F_n & A_1 \\ \dots & \\ F_2 & S_n IIprt \\ & A_2 \\ \dots & \\ F_1 & A_n \end{matrix} (***_{1.3})$$

where A_1 fits into A_2, ..., A_{n-1} fits into A_n, F_1 is forced out from F_2, …, F_{n-1} is forced out from F_n simultaneously and etc.

1.1 SIIprt - elements

Definition 1.1.1. The set of elements $\{a\} = (a_1, a_2, ..., a_n)$ containing to the set of elements $\{b\} = (b_1, b_2, ..., b_m)$ at one point x of space X we shall call SIIprt –

element, and such a point in space is called capacity of the SIIprt – element. We shall denote $SIIprt\underset{x}{\overset{\{a\}}{\{b\}}}$.

Definition 1.1.2. $SIIprt\underset{x}{\overset{\{a\}}{\{b\}}}$— dynamic-type set at x.

Definition 1.1.3. An ordered dynamic-type set of elements at one point in space is called an ordered SIIprt–element.

It's possible to $SIIprt\underset{x}{\overset{\{a\}}{\{b\}}}$ correspond to the set of elements $\{a\} \cup \{b\}$, $\{a\} \cap \{b\} = 0$, and the ordered SIIprt - 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 SIIprt – elements: $SIIprt\underset{x}{\overset{\{a\}}{\{b\}}} + SIIprt\underset{x}{\overset{\{q\}}{\{b\}}} = SIIprt\underset{x}{\overset{\{a\} \cup \{q\}}{\{b\}}}$.

The operator $SIIprt\underset{x}{\overset{\{a\} \cup \{q\}}{\{b\}}}$ is not equal $(\{a\} \cup \{q)\} \cup \{b\}$, rather, it is dynamic-type set at the point x. Similarly, for $SIIprt\underset{x}{\overset{\{a\} \cap \{q\}}{\{b\}}}$. This is more suitable for using sets for energy space, for any objects. The operator SIIprt is adapted for ordinary energies, using their property to overlap.

Capacity in itselfII

Definition 1.1.4. The capacity A in itselfII of the first type at point x is the capacity containing itselfII as an element at point x. Denote selfIIII(A) = $SII_1 f A = SIIprt\underset{x}{\overset{A}{A}}$.

Definition 1.1.5. The capacity A in itselfII of the second type at point x is the capacity that contains elements from which it can be generated. Denote $SII_2 f A$.

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

Definition 1.1.6. Partial capacity A in itselfII of the third type is the capacity A in itselfII as an element at point x, which partially contains itselfII or contains elements from which it can be generated in part or both. Let us denote $SII_3 f A$.

The following designations:

selfII$_1$(A) = $\overset{A}{SIIrt}\underset{A}{A}$, oselfII(A) = $\overset{x}{A}\underset{A}{SIIrt}$, oselfII$_1$(A) = $\overset{A}{A}\underset{A}{SIIrt}$, pselfII(A) = $\overset{x}{A}\underset{A}{SIIrt}\overset{A}{A}$, pselfII$_1$(A) = $\overset{A}{A}\underset{A}{SIIrt}\overset{A}{A}$.

Let us introduce the following notations: A*B = $\overset{A}{SIIrt}\underset{x}{B}$, A^2 = selfII(A) = $\overset{A}{SIIrt}\underset{x}{A}$, A^3 = selfII^2(A), …, A^{n+1} = selfII^n(A), … There is no commutativity here: A*B ≠ B*A . We can consider operator functions: $e^A = 1 + \frac{A}{1!} + \frac{A^2}{2!} + \frac{A^3}{3!} + \ldots$, $(A+B)^n = \sum_{k=0}^{n} \binom{n}{k} A^k B^{n-k}$, $(1+A)^n = 1 + \frac{Ax}{1!} + \frac{n(n-1)A^2}{2!} + \ldots$, etc.

You can consider a more “hard”option: A*B=$\overset{A}{PSIIrt}\underset{x}{B}$, where $\overset{A}{PSIIrt}\underset{x}{B}$– operator, containing A in every element of B at point x, A^2 = PselfII(A) =$\overset{A}{PSIIrt}\underset{x}{A}$, A^3 = PselfII^2(A), …, A^{n+1} = PselfII^n(A), … . There is no commutativity here: A*B ≠ B*A. We can consider operator functions: $e^A = 1 + \frac{A}{1!} + \frac{A^2}{2!} + \frac{A^3}{3!} + \ldots$, $(A+B)^n = \sum_{k=0}^{n} \binom{n}{k} A^k B^{n-k}$, $(1+A)^n = 1 + \frac{Ax}{1!} + \frac{n(n-1)A^2}{2!} + \ldots$, etc.

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

Connection of SIIprt – elements with capacities in themselves.

For example, $SIIrt_{g\{R\}}^{\{R\}}_{x}$ is the capacity in itselfII of the second type at point x if

$g\{R\}$ is a program capable of generating $\{R\}$.

Consider a third type of capacity in itselfII at point x. For example, based on

$SIIrt_{x}^{D}\{a\}$, where $\{a\} = (a_1, a_2, ..., a_n)$, i.e. n - elements at one point, we can consider

the capacity SII_3f in itselfII 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 structure $SIIrt_{x}^{D}\{a\}$ contains only m elements. If $D = Sprt_{x}^{\{a\}}$ we obtain

the structure SII_3f by form $(m^2, (n^2, 1))$, for m = 1, n = 2 the we have structure

SII_3f by form (1, (4, 1)).

Form (1.1) can be generalized into the following forms:

$$w^1_{m,n,k} = (k, (\ (...)\ ))^{(n_1,1)}_{(n_m,1)} \quad (1.1.1)$$

or

$$w^2_{m,n,k} = (k, (l, (...)))^{(n_1)}_{(n_m)} \quad (1.1.2)$$

$$w^3_{m,n,k,l} = Q((...), (\ (...)\ ))^{d_1\ \ (n_1,1)}_{d_l\ \ (n_m,1)} \quad (1.1.3),$$

where Q(x, y) – any operator, which makes a match between set $(...)^{d_1}_{d_l}$ and set $(\ (...)\ )^{(n_1,1)}_{(n_m,1)}$ or

$$w^4_{m,m_1,n_1,m_2,n_2,m_3,n_3} = (\mathrm{m}, ((m_1,n_1), ((m_2,n_2), (m_3,n_3)))) \ (1.1.4),$$

or

$$(\mathrm{Q}, \mathrm{R}) \ (1.1.5),$$

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 Srt, only by necessarily reducing the number of elements in the structure, in particular, using form

$$w_{m_1 \ldots}{}_{m_n} = (m_1,(m_2,(\ldots(m_n,1)\ldots))) \quad (1.2)$$

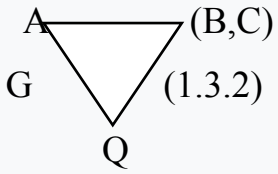
Structures more complex than S_3f can be introduced. For example, through a form that generalizes (1.1):

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

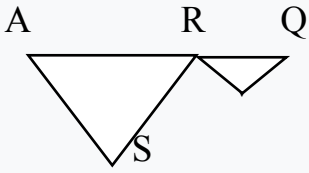
where A is compressed (fits) in C in the compression structure B in C (i.e., in the structure Srt^B_C); or

$w_{ABC} = A$ 

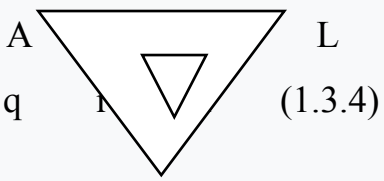
(1.3.1)



(1.3.2)



(1.3.3)



(1.3.4)

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

$$w_{A_1A_2\ldots A_nC} = (A_1,(A_2,(\ldots(A_n,C)\ldots))) \quad (1.4)$$

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 itselfII through a pseudo 3-connected form with a 2-connected form. The energy of the selfII-containment in itselfII closes on itselfII.

Math selfII

Let's consider SIIprt arithmetic first:

1. Simultaneous multiplication of $\{a\}$ and containing $\{c\}$ to $\{a\}$: $SIIprt\underset{x}{\overset{\{c\}}{\{a*\}}}$: the notation of the set B with elements $b_{i_1i_2\ldots i_n} = (SIIprt\underset{x}{\overset{\{c\}}{\left\{a_{1_{i_1}}*,a_{2_{i_2}}*,\ldots,\ a_{n_{i_n}}\right\}}})_R$ for any $\{i_1,i_2,\ldots,i_n\}$ without repetitions,x = $Sprt_w^{\{K\}}$,K-set of any $\{k_1*,k_2*,\ldots,k_n*\}$ without repeating them, k_i-any digit, i=1,2,…,n, R= $Sprt_w^{\{i_1+,i_2+,\ldots,i_n\}}$, 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 $Sprt_{\mathrm{x}}^{\{B+\}}$ 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 SIIprt-multiplication:

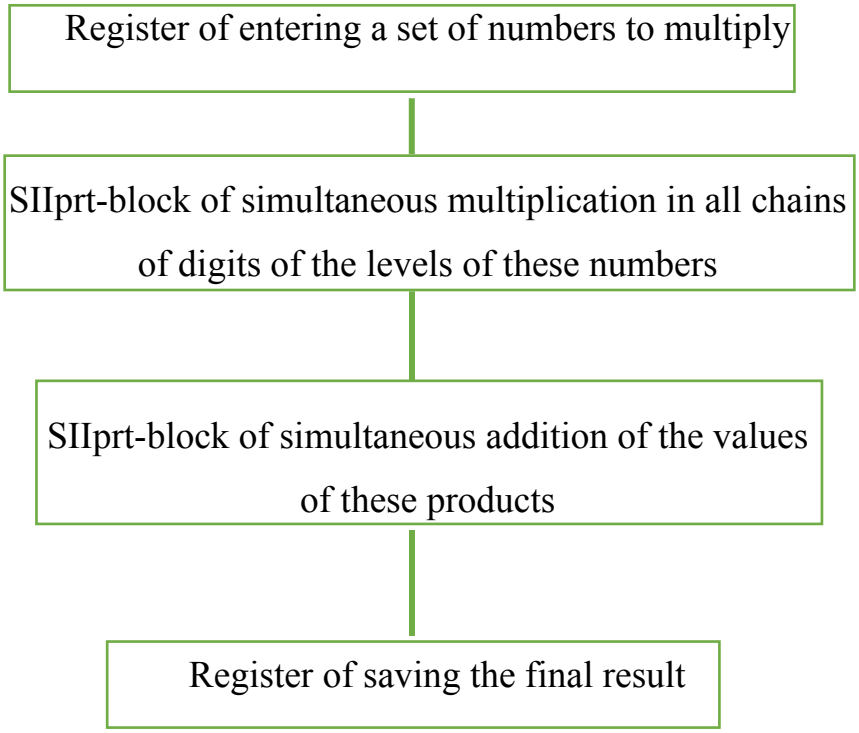


Fig. I.1. The straightforward functional scheme of the assumed arithmetic-logical device for SIIprt-multiplication.

Remark. 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.

2. Similarly, for simultaneous addition of a set of elements $\{a\} = (a_1,a_2,...,a_n)$ and containing $\{c\} = (c_1,c_2,...,c_m)$ to $\{a\}$ is carried out using $SIIprt^{\{c\}}_{x}\{a+\}$.

3. Similarly for simultaneous execution of various operations: $SIIprt^{\{c\}}_{x}\{aq\}$, where $\{q\} = (q_1,q_2,..,q_n)$. q_i -an operation, i = 1, …, n.

4. Similarly, for the simultaneous execution of various operators: $SIIprt^{\{c\}}_{x}\{Fa\}$, where $\{F\} = (F_1,F_2,...,F_n)$. F_i is an operator, i = 1, …, n.

5. The arithmetic itselfII for capacities in themselves will be similar: addition - $SII_1f\{a+\}$, (or $SII_3f\{a+\}$) for the third type), multiplication $SII_1f\{a*\}$, ($SII_3f\{a*\}$).

6. Similarly with different operations: $SII_1f\{aq\}$, ($SII_3f\{aq\}$), and with different operators: $SII_1f\{Fa\}$, ($SII_3f\{Fa\}$).

7. $SIIrt^{C}_{B}A -$ the result of the containment operator. For sets A, B, C we have

$$SIIrt^{C}_{B}A = \begin{Bmatrix} D \\ Q \\ A \cup B \cup C - A \cap B \cup (A \cap B \cap C) \end{Bmatrix},$$ where D is self²-set for $A \cap B \cap C$, Q is self-set for $A \cap B - A \cap B \cap C$. The measure:

$$m(SIIrt^{C}_{B}A) = \begin{pmatrix} \mu^{s^2}(A \cap B \cap C) \\ \mu^{s}(A \cap B - A \cap B \cap C) \\ \mu(A) + \mu(B) + \mu(C) - \mu(A \cap B) \end{pmatrix}.$$

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. SIIprt-double derivative of $f(x_1,x_2,\dots,x_n)$ is $SIIprt\left\{\frac{\partial}{\partial x_{1_i}},\frac{\partial}{\partial x_{2_i}},\dots,\frac{\partial}{\partial x_{k_i}}\right\}f(x_1,,x_2,\dots,x_n)$, where $x=(x_{1_i}, x_{2_i},\dots x_{k_i})$- any set from $(x_1,x_2,\dots,x_n)$. Let's designate SIIprt-$\frac{\partial^k f(x)}{\partial x_{1_i}\partial x_{2_i}\dots\partial x_{k_i}}$. SIIprt- double integral off $(x_1,x_2,\dots,x_n)$ is $\left\{\int()dx_{1_i},\int()dx_{2_i},\dots,\int()dx_{k_i}\right\}$ $SIIrt\left\{\int()dx_{1_i},\int()dx_{2_i},\dots,\int()dx_{k_i}\right\}f(x_1,,x_2,\dots,x_n)$, where $(x_{1_i}, x_{2_i},\dots x_{k_i})$- any set from $(x_1,x_2,\dots,x_n)$. Let's designate SIIprt-$\int\dots\int f(x)dx_{1_i}dx_{2_i}\dots dx_{k_i}$-k-multiple integral. SIIprt- double lim off $(x_1,x_2,\dots,x_n)$ is $\left\{\lim_{x_{1_i}\to b_{1_i}},\lim_{x_{2_i}\to b_{2_i}},\dots,\lim_{x_{k_i}\to b_{k_i}}\right\}$ $SIIrt\left\{\lim_{x_{1_i}\to a_{1_i}},\lim_{x_{2_i}\to a_{2_i}},\dots,\lim_{x_{k_i}\to a_{k_i}}\right\}f(x_1,,x_2,\dots,x_n)$. Let's designate SIIprt-$\lim_{\substack{x_{1_i}\to a_{1_i}\\ x_{1_i}\to b_{1_i}\\ \cdot\ \cdot\ \cdot\\ x_{k_i}\to a_{k_i}}} f(x_1,,x_2,\dots,x_n)$. SelfII-$\lim_{x\to a}=SIIrt\begin{matrix}\lim_{x\to a}\\ \lim_{x\to a}\\ \cdot\\ \lim_{x\to a}\end{matrix}$. Similarly, for SelfII-lim with n variables.
9. In the case of selfII-derivatives, inclusions of multiple derivatives are obtained. The same is true for selfII-integrals: we get inclusions of multiple integrals.
10. Let's denote selfII-(selfII-Q) through selfII^2-Q , fS(n,Q)= selfII-(selfII-(…(selfII-Q))) = selfII^n-Q for n-multiple selfII.

Operator itselfII

Definition 7. An operator that transforms $SIIrt\begin{matrix}\{c\}\\ \{a\}\\ x\end{matrix}$ into any $SII_i f\{b\}$,$i=2,3$;

where $\{b\}\subset\{a\}$; is the operator itselfII.

Example. The operator contains the set in itselfII.

In the case of lim-itselfII, for example, for m variables, it suffices to use the form (1.1) of lim-SIIprt 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).

About SIIprt and SII_3f programming.

The ideology of SIIprt and SII_3f can be used for programming. Here are some of the SIIprt programming operators [3]:

1. Simultaneous assignment of the expressions $\{p\} = (p_1,p_2,...,p_n)$ to the variables $\{\mathsf{a}\} = (\mathsf{a}_1,\mathsf{a}_2,...,\mathsf{a}_n)$ and simultaneous checking the set of conditions $\{f\} = (f_1, f_2, ...,f_n)$ for the set of expressions $\{B\} = (B_1, B_2, ...,B_n)$. This is implemented via $SIIprt_{IF}^{x}\{\{\mathsf{a}\} := \{p\}\}\{\{B\}\{f\}\}\ then\ Q$, where Q can be anything.
2. Similarly for loop operators and others.

SII_3f– software operators will differ only in that the aggregates $\{\mathsf{a}\},\{p\},\{B\},\{f\}$ will be formed from the corresponding SIIprt program operators in form (1.1) and for more complex operators in the form (1.2).

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.

Using SIIprt [3] we introduce the concept of SIIprt –change in physical quantity B with simultaneous checking: $SIIprt_{IF}^{x}\{\{\Delta_1 B,...,\Delta_n B\}\{\{\Delta_i B\}\{f\}\}\ then\ Q$. Then the mean SIIprt-velocity will be $v_{cpsIIt}(t, \Delta t) = SIIprt_{IF}^{x}\left\{\left\{\frac{\Delta_1 B}{\Delta t},...\frac{\Delta_n B}{\Delta t}\right\}\left\{\left\{\frac{\Delta_i B}{\Delta t}\right\}\{f\}\right\}\ then\ Q\right.$ and SIIprt-velocity at time t: $v_{sIIt} = \lim_{\Delta t\to 0} v_{cpsIIt}(t,\Delta t)$. SIIprt–acceleration $a_{sIIt} = \frac{dv_{sIIt}}{dt}$.

When using $SIIprt_x$ with "target weights", we get, depending on the "target weights", one or another modification, namely, for example, the velocity v^f_{sIIt}(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 $v^f_{sIIt} = SIIprt^{\{v_1 f, v_2\}}_x$, f – instantaneous replacement we get an instantaneous substitution $v_1 by\ v_2$ at point x of space at time t_0.

Consider, in particular, some examples: 1) $SIIprtIF\underset{x}{\overset{e}{\{spin(e)\}}}\ then\ Q$ 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 SIIprt elements.

Similarly, the concepts of SIIprt - force and SIIprt - energy are introduced. For example, $E^f_{sIIt} = SIIprt_{IF}\underset{x}{\overset{\{E_1 f, E_2\}}{\{\{E_i\}\{g\}\}}}\ then\ Q$ it would mean the instantaneous replacement of energy E_1 by E_2 at time t_0. Two aspects of SIIprt–energy should be distinguished: 1) carrying out the desired "target weight" and 2) fixing the result of it. Do not confuse energy - SIIprt (the node of energies) with SIIprt – 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.

Remark1.2. SIIprt – elements with "target weights" become peculiar. Here you need the necessary energy to carry them out. As a rule, this energy is at the level of SelfII. 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 selfII-energy of the object. For example, $SIIprt^{DNA}_{DNA}$ allows you to reach the level of DNA selfII-energy, $SIIprt^{Q}_{Q}$ allows you to reach the level of selfII-energy Q. The law of selfII-energy conservation operates already at the level of selfII-energy.

Also, in addition to capacities in themselves, you can consider the types of containment of oneselfII in oneselfII: the first type of the containment of oneselfII in oneselfII: the second type of the containment of oneselfII in oneselfII: potentially, for example, in the form of programming oneselfII, the third type is partial containment of oneselfII in themselves—for example, selfII-operator, selfII-action, whirlwind. A container containing itselfII can be formed by selfII-containment, i.e., containment in oneselfII. Let us clarify the concept of the term capacity in itselfII: it is a capacity containing itselfII potentially. Consider selfII-Q, where Q can be anything, including Q=selfII; in particular, it can be any action. Therefore, selfII-Q is when Q is made by itselfII; it makes itselfII. There is a partial selfII-Q for any Q with partial selfII-fulfillment. Let's consider several examples for capacities in themselves: ordinary lightning, electric arc discharge, and ball lightning.

oselfII-equation for x has its decision for x in direct kind.

oselfII-task for x has its decision for x in direct kind. oselfII-question has its answer for x in direct kind.

Definition 1.1.8. A structure with a second degree of freedom will be called complete, i.e., "capable" of reversing itselfII 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, $CIIprt\overset{A}{A} = SIIprtstr\overset{A}{A}$, $CIIrt\overset{A}{A}\ SIIrtstr\overset{A}{A}$ are such structures.

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

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

Remark 1.1.3. The concept of elements of physics SIIprt is introduced for energy space. The corresponding concept of elements of chemistry SIIprt is introduced accordingly. Examples: 1) $SIIprt_{A}^{IF\{a_1 q,a_2\}}\{\{a_i\}\{g\}\}\ then\ Q$ – the energy of instantaneous substitution a_1 by a_2 with checking, where a_1, and a_2 are chemical elements, q is instant replacement. Similarly, one can consider for the node of chemical reactions $SIIprt_{\text{checking}}^{\{chemical\ elements\ with\ "target\ weights"\}}$ or $SIIprt_{\text{checking}}^{\{chemical\ elements\ with\ "target\ weights"\}}$. The ideology of SIIprt elements allows us to go to the border of the world familiar to us, which allows us to act more effectively.

1.2 Dynamic SIIprt – elements

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

Definition 1.2.1. The process of fitting a set of elements $\{a(t)\} = (a_1(t),a_2(t),...,a_n(t))$ containing to the set of elements $\{b(t)\} = (b_1(t),b_2(t),...,b_m(t))$, which containing into one point x of space X at time t will be called a dynamic SIIprt – element. We will denote $SIIprt(t)_{x}^{\{a(t)\}}\{b(t)\}$.

Definition 1.2.2. Fitting ordered sets of elements into one point in space is called a dynamic ordered SIIprt–element.

It is allowed to sum dynamic SIIprt – elements:

$$SIIprt(t)_{x}^{\{a(t)\}}\{c(t)\} + SIIprt(t)_{x}^{\{b(t)\}}\{c(t)\} = SIIprt(t)_{x}^{\{a(t)\}\cup\{b(t)\}}\{c(t)\}.$$

Dynamic containment of oneselfII.

Definition 1.2.3. Dynamic SIIprt-capacity $SIIprt(\mathrm{t})\begin{matrix}\{R(t)\}\\ \{Q(t)\}\\ x\end{matrix}$ is the process of embedding $\{R(t)\}$ into $\{Q(t)\}$.

Definition 1.2.4. Dynamic SIIprt-capacity $\{A(t)\}$ is fitting into $itself II$ as an element of the zero types is the process of containing $\{A(t)\}$ itselfII as an element into one point x. Denote $SII_1f(t)\{A(t)\} = SIIprt(\mathrm{t})\begin{matrix}\{A(t)\}\\ \{A(t)\}\\ x\end{matrix}$.

Definition 1.2.5. The dynamic capacity $\{A(t)\}$containing itselfII as an element of the first type is the process of containing $\{A(t)\}$ itselfII as an element. Denote

$$SII_1f(t)\{A(t)\}= SIIprt(\mathrm{t})\begin{matrix}\{A(t)\}\\ \{A(t)\}\\ \{x(t)\}\end{matrix}.$$

Definition 1.2.6. Dynamic capacity C(t) in itselfII of the second type is the process of containing elements from which it can be generated. Let's denote $SII_2f(t)C(t)$.

Definition 1.2.7. Dynamic partial capacity B(t) in itselfII of the third type is a process of partial containment of B(t) in itselfII or elements from which it can be generated partially or both at the same time. Denote $SII_3f(t)B(t)$.

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

Connection of dynamic SIIprt – elements with dynamic containment of oneselfII. Consider third type of dynamic partial containment of oneselfII. For example, based on $SIIprt(\mathrm{t})\begin{matrix}\{a(t)\}\\ \{b(t)\}\\ x\end{matrix}$, where $\{a(t)\} = (\mathsf{a}_1(t),\mathsf{a}_2(t),...,\mathsf{a}_n(t))$, i.e. n – elements at one point x, we can consider the dynamic capacity in itselfII $SII_3f(t)$ 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 $Sprt(t)_{\mathsf{x}}^{\{\mathsf{a}(t)\}}$.

Dynamic containment of oneselfII of the third type can be formed for any other structure, not necessarily SIIprt, only through the obligatory reduction in the number of elements in the structure. In particular, using the form from (1.2) –(1.4). It is possible to introduce structures more complex than $SII_3f(t)$.

Dynamic math itselfII.

1. The process of simultaneous addition of a set of elements $\{\mathsf{a}(t)\} = (\mathsf{a}_1(t),\mathsf{a}_2(t),...,\mathsf{a}_n(t))$ and containing $\{c(t)\} = (c_1(t),c_2(t),...,c_m(t))$ to $\{a(t)\}$ are realized by $SIIprt(\mathrm{t})\{a(t) +\}_{x(t)}^{\{c(t)\}}$.
2. By analogy, for simultaneous multiplication: $SIIprt(\mathrm{t})\{a(t) *\}_{x(t)}^{\{c(t)\}}$.
3. Similarly for simultaneous execution of various operations: $SIIprt(\mathrm{t})\{a(t)\, q(t)\}_{x(t)}^{\{c(t)\}}$, where $\{q(t)\} = (q_1(t),\mathsf{q}_2(t),..,\mathsf{q}_n(t))$. $\mathsf{q}_i(t)$-an operation, i = 1, …, n.
4. Similarly, for the simultaneous execution of various operators: $SIIprt(\mathrm{t})\{F(t)a(t)\}_{x(t)}^{\{c(t)\}}$, where $\{F(t)\} = (F_1(t),F_2(t),...,F_n(t))$. $\mathrm{F_i(t)}$ is an operator, i = 1,…,n.
5. Dynamic arithmetic itselfII for containments of oneselfII will be similar: dynamic addition - $SII_1f(t)\{a(t) +\}$, (or $SII_3f(t)\{a(t) +\}$ for the third type), dynamic multiplication $SII_1f(t)\{a(t) *\}$, ($SII_3f(t)\{a(t) *\}$).
6. Similarly with different operations: $SII_1f(t)\{\mathsf{a}(t)q(t)\}$, ($SII_3f(t)\{\mathsf{a}(t)q(t)\}$) and with different operators: $SII_1f(t)\{F(t)\mathsf{a}(t)\}$, ($SII_3f(t)\{F(t)\mathsf{a}(t)\}$).

7. $SIIrt(\mathrm{t})\overset{C(t)}{\underset{B(t)}{A(t)}}$ – the result of the containment operator. For sets A(t), B(t), C(t) we have

$$SIIrt(\mathrm{t})\overset{C(t)}{\underset{B(t)}{A(t)}}=\left\{\begin{matrix} D(t) \\ Q(t) \\ A(t)\cup B(t)\cup C(t)-A(t)\cap B(t)\cup(A(t)\cap B(t)\cap C(t)) \end{matrix}\right\},$$

where D(t) is self^2-set for $A(t)\cap B(t)\cap C(t)$, Q is self-set for $A(t)\cap B(t)-A(t)\cap B(t)\cap C(t)$. The measure:

$$\mathrm{m}(SIIrt(\mathrm{t})\overset{C(t)}{\underset{B(t)}{A(t)}})=\begin{pmatrix} \mu^{s^2}(A(t)\cap B(t)\cap C(t)) \\ \mu^{s}(A(t)\cap B(t)-A(t)\cap B(t)\cap C(t)) \\ \mu(A(t))+\mu(B(t))+\mu(C(t))-\mu(AA(t)\cap B(t)) \end{pmatrix}.$$

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 SIIprt(t)-derivatives, dynamic SIIprt(t)-integrals, dynamic SIIprt(t)-lim, dynamic selfII(t)-derivatives, dynamic selfII-integrals.
9. Denote dynamic selfII-(dynamic selfII-Q(t)) through dynamic selfII^2-Q(t) , fS(t)(n,Q(t))= dynamic selfII-(dynamic selfII-(…(dynamic selfII-Q(t)))) = dynamic selfII^n-Q(t) for n-multiple dynamic selfII.

Remark 1.2.1. We consider expression

$$\begin{matrix} C(\mathrm{t}) & A(\mathrm{t}) \\ D(\mathrm{t})SIIprt(\mathrm{t}) & B(\mathrm{t}) \\ F(\mathrm{t}) & Q(\mathrm{t}) \end{matrix} \quad (*_{2.1}),$$

where A(t) fits into B(t), B(t) fits into Q(t), F(t) is forced out from D(t), D(t) is forced out from C(t) simultaneously. A(t), B(t), D(t), C(t), F(t), Q(t) are any, in particular, structures. The result of this process will be described by the expression

$$\begin{matrix} C(\mathrm{t}) & & A(\mathrm{t}) \\ D(\mathrm{t}) & SIIrt(\mathrm{t}) & B(\mathrm{t}) \quad (*_{2.2}) \\ F(\mathrm{t}) & & Q(\mathrm{t}) \end{matrix}$$

If A(t), B(t), D(t), C(t), F(t), Q(t) are taken as sets, then we will call $(*_{2.1})$ a dynamic set.

It can be considered a simpler version of the dynamic set

$$\begin{matrix} & A(\mathrm{t}) & \\ SIIprt(\mathrm{t}) & B(\mathrm{t}) & (**_{2.1}), \\ & Q(\mathrm{t}) & \end{matrix}$$

where set A(t) fits into set B(t), B(t) fits into Q(t) simultaneously, the result of this process will be described by the expression

$$\begin{matrix} & A(\mathrm{t}) & \\ SIIrt(\mathrm{t}) & B(\mathrm{t}) & (**_{2.2}) \\ & Q(\mathrm{t}) & \end{matrix}$$

or

$$\begin{matrix} C(\mathrm{t}) \\ B(\mathrm{t}) \quad SIIprt(\mathrm{t}) (***_{2.1}) \\ A(\mathrm{t}) \end{matrix}$$

where set A(t) is forced out from B(t), B(t) is forced out from C(t), the result of this process will be described by the expression

$$\begin{matrix} C(\mathrm{t}) \\ B(\mathrm{t}) \quad SIIrt(\mathrm{t}) \quad (***_{2.2}) \\ A(\mathrm{t}) \end{matrix}$$

$$\mathrm{oselfII}(A(\mathrm{t})) = \begin{matrix} A(\mathrm{t}) \\ A(\mathrm{t}) SIIprt(\mathrm{t}) \\ A(\mathrm{t}) \end{matrix}$$ denotes the dynamic expelling A(t) oneselfII out of oneselfII, The following designations:

$$\text{selfII}_1(\text{A(t)}) = SIIrt\,(\text{t})\begin{matrix}A\,(\text{t})\\A\,(\text{t})\\A\,(\text{t})\end{matrix}, \text{oselfII(A(t))} = \begin{matrix}x\,(\text{t})\\A\,(\text{t})\\A\,(\text{t})\end{matrix}SIIrt\,(\text{t}), \text{oselfII}_1(\text{A(t)}) =$$

$$\begin{matrix}A\,(\text{t})\\A\,(\text{t})\\A\,(\text{t})\end{matrix}SIIrt\,(\text{t}), \text{pselfII(A(t))} = \begin{matrix}x\,(\text{t})\\A\,(\text{t})\\A\,(\text{t})\end{matrix}SIIrt\,(\text{t})\begin{matrix}A\,(\text{t})\\A\,(\text{t})\\x\,(\text{t})\end{matrix}, \text{pselfII}_1(\text{A(t)}) =$$

$$\begin{matrix}A\,(\text{t})\\A\,(\text{t})\\A\,(\text{t})\end{matrix}SIIrt\,(\text{t})\begin{matrix}A\,(\text{t})\\A\,(\text{t})\\A\,(\text{t})\end{matrix}.$$

pselfII(A(t)), pselfII$_1$(A(t)) - simultaneous dynamic containment A(t) of oneselfII in oneselfII and dynamic expelling A(t) oneselfII out of oneselfII. oselfII(A(t)), oselfII$_1$(A(t)) will be called dynamic anti-capacities from oneselfII. 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 itselfII. The energy of selfII-containment in itselfII is closed on itselfII [5].

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

About dynamic SIIprt and SII$_3$f(t) programming.

The ideology of dynamic SIIprt and SII$_3$f(t) can be used for programming:

1. The process of simultaneous assignment of the expressions $\{p(t)\} = (p_1(t),p_2(t),...,p_n(t))$ to the variables $\{g(t)\} = (g_1(t),g_2(t),...,g_n(t))$ and simultaneous checking the set of conditions $\{f(t)\} = (f(t)_1, f_{2,}(t) ...,f(t)_n)$ for the set of expressions $\{B(t)\} = (B_1(t), B_2(t), ...,B_n(t))$. is implemented through

$$SIIprt(t)\ IF\{\{B(t)\}\{f(t)\}\}\ then\ Q(t), \text{ where Q(t) can be any..}$$

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

2. Similarly for loop operators and others.

$SII_3f(t)$– software operators will differ only in that the aggregates $\{g(t)\},\{p(t)\},\{B(t)\},\{f(t)\}$ will be formed from corresponding processes SIIprt(t) for the above-mentioned programming operators through form (1.1) or form (1.2) for more complex operators.

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

$$E^f_{sIIt}(t) = SIIprt(t)IF\{\{E_i(t)\}\{g(t)\}\}\ then\ Q(t)$$

$$\{E_1(t)f(t),E_2(t)\}$$
$$x(t)$$

will mean the process of instantaneous replacement f of energy $E_1(t)$ by $E_2(t)$ at time t. Similarly, using $SII_if(t)$, the concepts of SII_if(t)-force, SII_if(t)-energy, i=1,2,3, and etc are introduced.

Remark 1.2.3. It is the containment of oneselfII in oneselfII that can “give birth” to the capacities in itselfII – that is what selfII-organization is.

$$B(t)$$
$$SIIprt(t)B(t)$$
$$B(t)$$
$$B(t)$$

Remark 1.2.4. $SIIprt(t)SIIprt(t)B(t)$ can increase selfII- level of B(t).

$$B(t)$$
$$B(t)$$
$$SIIprt(t)B(t)$$
$$B(t)$$

Remark 1.2.5. For example, the operator itselfII is $SII_1f(t)$.

Remark 1.2.6. May be considered the following derivatives: $\frac{dSIIprt(t)C(t)^{A(t)}_{B(t)}}{dt}$, $\frac{d\,C(t)SIIprt(t)._{B(t)}^{A(t)}}{dt}$, $\frac{d\,R(t)SIIprt(t)C(t)_{P(t)}^{Q(t)}{}_{B(t)}^{A(t)}}{dt}$, $\frac{dSII_if(t)}{dt}$, i= 0, 1, 2, 3.

Remark 1.2.7. It is the containment of oneselfII in itselfII as an element that can be interpreted as dynamic capacities in itselfII.

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

1.3 SIIprt – elements for continual sets

Earlier, we considered finite-dimensional discrete SIIprt-elements and selfII-capacities in itselfII as an element. Here we believe some continual SIIprt-elements and continual selfII-capacities in themselves as an element [3].

Definition 1.3.1. The set of continual elements $\{a\} = (a_1,a_2,...,a_n)$ containing to the set of elements $\{b\} = (b_1,b_2,...,b_m)$ at one point x of space X will be called continual SIIprt – element, and such a point in space will be called capacity of the continual SIIprt – element. We will denote $SIIprt_{x}^{\{a\}}\{b\}$.

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

It's allowed to sum continual SIIprt – elements: $SIIprt_{x}^{\{a\}}\{b\} + SIIprt_{x}^{\{q\}}\{b\} = SIIprt_{x}^{\{a\}\cup\{q\}}\{b\}$, where some or any elements may be ordered elements.

Definition 1.3.3. The continual selfII-capacity A in itselfII as an element of the first type is the capacity containing itselfII as an element. Denote SII_1fA.

Definition 1.3.4. The ordered continual selfII-capacity A in itselfII as an element of the first type is the ordered capacity containing itselfII as an element. Denote $\overrightarrow{SII_1 f A}$.

Definition 3.5. The continual selfII-capacity A in itselfII, as an element of the second type, is the capacity containing elements from which it can be generated. Let's denote $SII_2 f A$.

An example of continual selfII-capacity in itselfII 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 selfII-capacity in itselfII as an element of the third type is called continual selfII-capacity in itselfII as an element that partially contains itselfII or contains elements from which it can be generated in part or both simultaneously. Denote $SII_3 f$.

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

The connection of continual SIIprt – elements with continual selfII-capacities in themselves as an element.

Consider a third type of continual selfII-capacity in itselfII as an element. For

example, based on $Sprt_{\mathsf{x}}^{\{\mathsf{a}\}}$, where $\{\mathsf{a}\} = (\mathsf{a}_1,\mathsf{a}_2,...,\mathsf{a}_n)$, i.e. a_i - continual elements at one point, i=1, 2, …, n. The continual selfII-capacity in itselfII as an element with m continual elements from $\{\mathsf{a}\}$, at m<n, can be considered as SII_3f, which is formed by the form (1.1), i.e., only m continual elements are located in the structure $Sprt_{\mathsf{x}}^{\{\mathsf{a}\}}$. Continual selfII-capacities in itselfII as an element of the third type can be formed for any other structure, not necessarily Sit, only by obligatory reducing the number of continual elements in the structure. In particular, using the form (1.2). Structures more complex than S_3f can be introduced.

Mathematics itselfII for continual elements.

1.Simultaneous addition of the continual elements of the set $\{a\} = (\mathsf{a}_1,\mathsf{a}_2,...,\mathsf{a}_n)$ and containing $\{c\} = (c_1,c_2,...,c_m)$ to $\{a\}$ is carried out using $SIIprt_{x}^{\{c\}}\{a \cup\}$.

2. By analogy, for simultaneous multiplication: $SIIprt_{x}^{\{c\}}\{a \cap\}$.

3. Similarly, for simultaneous execution of various operations: $SIIprt_{x}^{\{c\}}\{aq\}$,,

where $\{q\} = (q_1,\mathsf{q}_2,..,\mathsf{q}_\mathrm{n})$. q_i -an operation, i = 1, …, n.

4. Similarly, for the simultaneous execution of various operators: $SIIprt_{x}^{\{c\}}\{Fa\}$,

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

5. For continual selfII-capacities in themtselfII as an element will be similar: addition - $SII_1f\{a+\}$, (or $SII_3f\{a+\}$ for the third type), multiplication $SII_1f\{a*\}$, ($SII_3f\{a*\}$).

6. Similarly with different operations: $SII_1f\{\mathsf{a}q\}$, ($SII_3f\{\mathsf{a}q\}$), and with different operators: $SII_1f\{F\mathsf{a}\}$, ($SII_3f\{F\mathsf{a}\}$).

7.$SIIrt_{B}^{C}A$ – the result of the containment operator. For continual sets A, B, C we have

$$SIIrt\begin{matrix} C \\ A \\ B \end{matrix} = \left\{ \begin{matrix} D \\ Q \\ A \cup B \cup C - A \cap B \cup (A \cap B \cap C) \end{matrix} \right\}$$, where D is self²-set for $A \cap B \cap C$, Q is self-set for $A \cap B - A \cap B \cap C$. The measure:

$$m(SIIrt\begin{matrix} C \\ A \\ B \end{matrix}) = \begin{matrix} \mu^{s^2}(A \cap B \cap C) \\ \mu^{s}(A \cap B - A \cap B \cap C) \\ \mu(A) + \mu(B) + \mu(C) - \mu(A \cap B) \end{matrix}.$$

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.

These elements are used for SIIprt-coding, SIIprt translation, coding selfII, and translation selfII 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-- Selb-space. Then we introduce the scalar product for functions or vectors and get the Hilbert space. We call this space Selh-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].

1.4 Dynamic continual SIIprt – elements

Definition 1.4.1. . The process of fitting a set of continual elements $\{a(t)\} = (a_1(t), a_2(t), ..., a_n(t))$ containing to the set of elements $\{b(t)\} =$

$(b_1(t),b_2(t),...,b_m(t))$, which containing into one point x of space X at time t will be called a dynamic continual SIIprt – element. We will denote $SIIprt(\mathrm{t})\begin{matrix}\{a(t)\}\\ \{b(t)\}\\ x\end{matrix}$.

Definition 1.4.2. The process of containing an ordered set of continual elements at one point in space is called dynamic continual ordered SIIprt – element.

It is allowed to sum dynamic continual SIIprt – elements:

$$SIIprt(\mathrm{t})\begin{matrix}\{a(t)\}\\ \{c(t)\}\\ x\end{matrix} + SIIprt(\mathrm{t})\begin{matrix}\{b(t)\}\\ \{c(t)\}\\ x\end{matrix} = SIIprt(\mathrm{t})\begin{matrix}\{a(t)\}\cup\{b(t)\}\\ \{c(t)\}\\ x\end{matrix}.$$

Dynamic continual containment of oneselfII in oneselfII as an element.

Definition 1.4.3. Dynamic continual SIIprt-capacity $SIIprt(\mathrm{t})\begin{matrix}\{R(t)\}\\ \{Q(t)\}\\ x\end{matrix}$ is the process of embedding $\{R(t)\}$ into $\{Q(t)\}$.

Definition 1.4.4. Dynamic continual SIIprt-capacity $\{A(t)\}$is fitting into $\{A(t)\}$as an element of the zero types is the process of containing $\{A(t)\}$ into itselfII as an element into one point x. Denote $SII_1f(t)\{A(t)\} = SIIprt(\mathrm{t})\begin{matrix}\{A(t)\}\\ \{A(t)\}\\ x\end{matrix}$.

Definition 1.4.5. The dynamic capacity continual $\{A(t)\}$containing itselfII as an element of the first type is the process of containing $\{A(t)\}$ into itselfII as an element. Denote $SII_1f(t)\{A(t)\}= SIIprt(\mathrm{t})\begin{matrix}\{A(t)\}\\ \{A(t)\}\\ \{x(t)\}\end{matrix}$.

Definition 1.4.6. The dynamic containment continual C(t) of oneselfII of the second type embedding contains the continual elements from which it can be generated. Denote $SII_2f(t)C(t)$.

Definition 1.4.7. The partial dynamic containment continual B(t) of oneselfII of the third type is the process of partial embedding continual B(t) into oneselfII or

continual elements from which it can be generated in part or both simultaneously. Denote $SII_3f(t)B(t)$.

The connection of dynamic continual SIIprt – elements with dynamic containment of oneselfII in oneselfII as an element [3].

Let us consider the partial dynamic continual containment of oneselfII in oneselfII as an element of the third type. For example, based on $SIIprt(\mathrm{t})\underset{x}{\{a(t)\}}\{b(t)\}$, where $\{\mathsf{a}(t)\} = (\mathsf{a}_1(t),\mathsf{a}_2(t),...,\mathsf{a}_n(t))$, i.e. n - continual elements at one point x, one can consider the dynamic containment $SII_3f(t)$ of oneselfII in oneselfII as an element with m continual elements from $\{\mathsf{a}(t)\}$, m<n, which is a process that is necessary form according to the form (1.1), i.e., only m continual elements from $\{\mathsf{a}(t)\}$ are located in the structure $SIIprt(\mathrm{t})\underset{x}{\{a(t)\}}\{b(t)\}$. Dynamic continual containments of oneselfII in oneselfII as an element of the third type can be formed for any other structure, not necessarily SIIprt, only by necessarily reducing the number of continual elements in the structure. In particular, with the help of form (1.1.1) – (1.4). It is possible to introduce structures more complex than $S_3f(t)$.

Dynamic continual mathematics selfII.

2. The process of simultaneous addition of a set of continual elements $\{\mathsf{a}(t)\} = (\mathsf{a}_1(t),\mathsf{a}_2(t),...,\mathsf{a}_n(t))$ and containing $\{c(t)\} = (c_1(t),c_2(t),...,c_m(t))$ to $\{a(t)\}$ are realized by $SIIprt(\mathrm{t})\underset{x(t)}{\{a(t)\cup\}}\{c(t)\}$.

1.

2. By analogy, for simultaneous multiplication: $SIIprt(\mathrm{t})\underset{x(t)}{\{a(t)\cap\}}\{c(t)\}$.

3. Similarly for simultaneous execution of various operations:

$SIIprt(\mathrm{t})\{a(t)\,q(t)\}$, where $\{q(t)\} = (q_1(t), q_2(t), .., q_n(t))$. $q_i(t)$-an operation, i = 1, …, n.

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

$SIIprt(\mathrm{t})\{F(t)a(t)\}$, where $\{F(t)\} = (F_1(t), F_2(t), ..., F_n(t))$. $F_i(t)$ is an operator, i = 1, …, n.

5. The dynamic arithmetic selfII for the dynamic continual containments of oneselfII will be similar: dynamic addition - $SII_1 f(t)\{a(t) \cup\}$, (or $SII_3 f(t)\{a(t) \cup\}$ for the third type), dynamic multiplication $SII_1 f(t)\{a(t) \cap\}$, ($SII_3 f(t)\{a(t) \cap\}$).

6. Similarly with different operations: $SII_1 f(t)\{\mathsf{a}(t)q(t)\}$, ($SII_3 f(t)\{\mathsf{a}(t)q(t)\}$), and with different operators: $SII_1 f(t)\{F(t)\mathsf{a}(t)\}$, ($SII_3 f(t)\{F(t)\mathsf{a}(t)\}$).

7. $SIIrt(\mathrm{t})A(t)$ – the result of the containment operator. For continual sets $\begin{matrix} C(t) \\ B(t) \end{matrix}$ A(t), B(t), C(t) we have

8. $$SIIrt(\mathrm{t})A(t) = \left\{ \begin{matrix} D(t) \\ Q(t) \\ A(t) \cup B(t) \cup C(t) - A(t) \cap B(t) \cup (A(t) \cap B(t) \cap C(t)) \end{matrix} \right\},$$

where D(t) is self[2]-set for $A(t) \cap B(t) \cap C(t)$, Q is self-set for $A(t) \cap B(t) - A(t) \cap B(t) \cap C(t)$. The measure:

9. $$\mathsf{m}(SIIrt(\mathrm{t})A(t)) = \left(\begin{matrix} \mu^{s^2}(A(t) \cap B(t) \cap C(t)) \\ \mu^{s}(A(t) \cap B(t) - A(t) \cap B(t) \cap C(t)) \\ \mu(A(t)) + \mu(B(t)) + \mu(C(t)) - \mu(AA(t) \cap B(t)) \end{matrix} \right).$$

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.

10. Similarly, for dynamic SIIprt-derivatives, dynamic SIIprt-integrals, dynamic SIIprt-lim, dynamic selfII-derivatives, dynamic selfII-integrals.
11. Denote dynamic continual selfII-(dynamic continual selfII-Q(t)) through dynamic continual $selfII^2$-Q(t), fS(t)(n,Q(t))= dynamic continual selfII-(dynamic continual selfII-(...(dynamic continual selfII-Q(t)))) = dynamic continual $selfII^n$-Q(t) for n-multiple dynamic continual selfII.

Remark 1.4. We consider expression

$$\begin{array}{ccc} C(t) & & A(t) \\ D(t) & SIIprt(t) & B(t) \\ F(t) & & Q(t) \end{array} \quad (*_{2.1}),$$

where continual A(t) fits into continual B(t), continual B(t) fits into continual Q(t), continual F(t) is forced out from continual D(t), continual D(t) is forced out from continual C(t) simultaneously. A(t), B(t), D(t), C(t), F(t), Q(t) are any, in particular, structures. The result of this process will be described by the expression

$$\begin{array}{ccc} C(t) & & A(t) \\ D(t) & SIIrt(t) & B(t) \\ F(t) & & Q(t) \end{array} \quad (*_{2.2})$$

If A(t), B(t), D(t), C(t), F(t), Q(t) are taken as sets, then we will call $(*_{2.1})$ a dynamic continual set.

It can be considered a simpler version of the dynamic continual set

$$\begin{array}{ccc} & & A(t) \\ SIIprt(t) & & B(t) \\ & & Q(t) \end{array} \quad (**_{2.1}),$$

where continual set A(t) fits into continual set B(t), continual B(t) fits into continual Q(t) simultaneously, the result of this process will be described by the expression

$$\begin{matrix} A(\mathrm{t}) \\ SIIrt(\mathrm{t})\, B(\mathrm{t}) \\ Q(\mathrm{t}) \end{matrix} \quad (**_{2.2})$$

or

$$\begin{matrix} C(\mathrm{t}) \\ B(\mathrm{t})\, SIIprt(\mathrm{t}) \\ A(\mathrm{t}) \end{matrix} \quad (***_{2.1})$$

where continual set A(t) is forced out from continual B(t), continual B(t) is forced out from continual C(t), the result of this process will be described by the expression

$$\begin{matrix} C(\mathrm{t}) \\ B(\mathrm{t})\, SIIrt(\mathrm{t}) \\ A(\mathrm{t}) \end{matrix} \quad (***_{2.2})$$

$$\text{oselfII}(A(\mathrm{t})) = \begin{matrix} A(\mathrm{t}) \\ A(\mathrm{t})\, SIIprt(\mathrm{t}) \\ A(\mathrm{t}) \end{matrix}$$ denotes the dynamic expelling continual A(t)

oneselfII out of oneselfII, The following designations:

$$\text{selfII}_1(\text{A(t)}) = SIIrt\,(\mathrm{t}) \begin{matrix} A(\mathrm{t}) \\ A(\mathrm{t}) \\ A(\mathrm{t}) \end{matrix}, \quad \text{oselfII(A(t))} = \begin{matrix} x(\mathrm{t}) \\ A(\mathrm{t})\, SIIrt\,(\mathrm{t}) \\ A(\mathrm{t}) \end{matrix}, \quad \text{oselfII}_1(\text{A(t)}) =$$

$$\begin{matrix} A(\mathrm{t}) \\ A(\mathrm{t})\, SIIrt\,(\mathrm{t}) \\ A(\mathrm{t}) \end{matrix}, \quad \text{pselfII(A(t))} = \begin{matrix} x(\mathrm{t}) \\ A(\mathrm{t})\, SIIrt\,(\mathrm{t}) \\ A(\mathrm{t}) \end{matrix}, \quad \text{pselfII}_1(\text{A(t)}) =$$

$$\begin{matrix} A(\mathrm{t}) & A(\mathrm{t}) \\ A(\mathrm{t})\, SIIrt\,(\mathrm{t}) & A(\mathrm{t}). \\ A(\mathrm{t}) & A(\mathrm{t}) \end{matrix}$$

Connection of dynamic continual SIIprt – elements with target weights with dynamic continual containment of oneselfII with target weights [3].

Consider a third type of dynamic partial containment of oneselfII with target

weights g(t) [3]. For example, based on $SIIprt(\mathrm{t})\{a(t)\mathrm{g}(\mathrm{t})\}$, where $\{a(t)\} = \begin{matrix}\{c(t)\}\\ x(t)\end{matrix}$ $(a_1(t),a_2(t),...,a_n(t))$, i.e. n - continual elements with target weights $\{\mathrm{g}(\mathrm{t})\}$ at one point x, we can consider the dynamic containment $SII_3 f(t)a(t)\mathrm{g}(\mathrm{t})$ of oneselfII with target weights with m continual elements with target weights $\{\mathrm{g}(\mathrm{t})\}$ 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 $\{\mathrm{g}(\mathrm{t})\}$ from $\{a(t)\}$ are located in the structure $SII_3 f(t)a(t)\mathrm{g}(\mathrm{t})$. Dynamic containments of oneselfII 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 (1.1.1) – (1.4). Structures more complex than $SII_3 f(t)a(t)\mathrm{g}(\mathrm{t})$ can be introduced.

Definition 1.4.8. Dynamic continual SIIprt-capacity $\{A(t)\}$is fitting into itselfII as an element with target weights $\{\mathrm{g}(\mathrm{t})\}$ of the zero types is the process of $\{A(t)\}$ containing itselfII as an element with target weights $\{\mathrm{g}(\mathrm{t})\}$ into one point x.

Denote $SII_1 f(t)\{A(t)\}\mathrm{g}(\mathrm{t}). = SIIprt(\mathrm{t})\begin{matrix}\{A(t)\mathrm{g}(\mathrm{t})\}\\ x\end{matrix}\{A(t)\mathrm{g}(\mathrm{t})\}$.

Definition 30. The dynamic containment of continual C(t) itselfII into itselfII with target weights $\{\mathrm{g}(\mathrm{t})\}$ of the second type is the process of containment of the continual elements from which it can be generated. Let's denote $SII_2 f(t)C(t)\mathrm{g}(\mathrm{t})$.

Definition 1.4.9. Partial dynamic containment of continual B(t) itselfII into itselfII with target weights $\{\mathrm{g}(\mathrm{t})\}$ of the third type is the process of partial containment of continual B(t) into itselfII or continual elements from which it can be generated partially, or both at the same time. Denote $SII_3 f(t)B(t)\mathrm{g}(\mathrm{t})$.

1.5 The usage of SIIprt-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 [3]. 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 SIIprt-networks. SIIprt-networks (SmnSIIprt) are a SIIprt-structure that can be built for the required weights. SIIprt-OS (SIIprt operating system) uses SIIprt-coding and SIIprt-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. SIIprt-coding (or selfII-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. SIIprt-translation is carried out by inversion. In this case, selfII-coding and selfII-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 [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 SIIprt (designation - mnSIIprt) for simple information processing. More complex executing programs are used for mnSIIprt nodes. The first level of mnSIIprt consists of simple mnSIIprt. The second level of mnSIIprt consists of $SIIprt(t)^{\{mnSIIprt\}}_{D}$ – SIIprt-node of mnSIIprt

in range D, D- capacity for mnSIIprt node. The third level of mnSIIprt consists of

$$Sprt_{D}^{\{St_{D}^{\{mnSt\}}\}}\ SIIprt(t)\begin{matrix}\{mnSIIprt\}\\ SIIprt(t)\{mnSIIprt\}\\ D\\ \{mnSIIprt\}\\ SIIprt(t)\{mnSIIprt\}\\ D\\ D\end{matrix}$$ - SIIprt^2- node of mnSIIprt in range D,

thus D becomes capacity of itselfII in itselfII as an element for mnSIIprt. For our networks, it is sufficient to use SIIprt^2- nodes of mnSIIprt, but selfII-level is higher in living organisms, particularly SIIprt^n-, n≥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 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. mnSIIprt contains

$$SIIprt(t)\begin{matrix}\{eIprograms\}\\ \{mnSIIprt\}\\ \{mnSIIprt\}\end{matrix}$$, eIprogram −executing program in SIIprt- OS. SIIprt-OS (or SelfII-OS) is based on SIIprt-assembly language (or SelfII-assembly language), which is based on assembly language through SIIprt-approach in turn, if the base of elements of SIIprt-networks is sufficient. The eprograms are in SIIprt-programming environments (or SelfII-programming environments), but this question and SIIprt-networks base will be considered in the following monographs. In particular, eprograms may contain SIIprt- programming operators. In mnSIIprt cores, the constant memory SIIprt with correspondent eprograms depending on mnSIIprt.

The OS (operating system) and the principles and modes of operation of the SIIprt-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 SIIprt – physics and special neural networks with artificial neurons operating in normal and SIIprt-modes. Let's denote this model through SmnSIIprt. To do this, it's proposed to use mnSIIprt of different levels; for example, for the usual mode, mnSIIprt 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 SIIprt–mode with the desired "target weight" is realized in this section, etc., to the center. In the case of a monster during the test, SmnSIIprt is activated with the desired "target weight." Here are realized other tasks also. To reach the selfII-

{mnSIIprt}

energy level, the mode *SIIprt*(t){mnSIIprt} is used. In normal mode, it's planned

{mnSIIprt}

to carry out the movement of SmnSIIprt 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 SmnSIIprt 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 SmnSIIprt. SmnSIIprt is represented by a neural network that extends from the center of one of the main clusters of SIIprt - artificial neurons to the shell, turning into the body itselfII. 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 SmnSIIprt's actions is below the operator's cab. In SIIprt – mode, the entire network or its sections are SIIprt – activated to perform specific tasks, in particular, with "target weights." In the target, block used Sit-coding, SIIprt-translation for activation of all networks to "target weights" simultaneously, then –the reset of this SIIprt-coding after activation. Unfortunately, triodes are not suitable for SIIprt -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 SIIprt-operative memory belt is disposed around a central core of SmnSIIprt. There are SIIprt-coding, SIIprt-translation, and SIIprt-realize of eprograms and the programs from the archives without extraction, SIIprt-coding and SIIprt-translation may be used in high-intensity, ultra-short optical pulses laser of Nobel laureates 2018-year Gerard Mourou, Donna, Strickland. SIIprt – structure or an eprogram if one is present of needed «target weight» are taken in target block at SIIprt – activation of the networks. $SIIprt(\mathrm{t})^{\mathrm{Smn_{SIIprt}},f}_{\mathrm{Smn_{SIIprt}},f\ a\mathit{ctivation}}$ derives SmnSIIprt to the selfII-level boundary with target weight f.

It's used an alternating current of above high frequency and ultra-violet light, which can work with SIIprt – structures in SIIprt–modes by its nature to activate the networks or some of its parts in SIIprt–modes and locally using SIIprt–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 [10-21] 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, itselfII, 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,

$$\begin{matrix} C & A \\ DSIIprt(t) & B \\ F & Q \end{matrix} = \begin{cases} \begin{matrix} C \\ DSIIprt, \\ F \end{matrix} & q_2 \geq t \geq q_1 \\ \begin{matrix} Q & A \\ DSIIprt & B \\ F & Q \end{matrix} & q_3 \geq t > q_2 \\ \begin{matrix} C & A \\ DSIIprt & B \\ F & Q \end{matrix} & q_4 \geq t > q_3 \quad (*_{6.1}), \\ \begin{matrix} A \\ SIIprt \\ Q \end{matrix} B & q_5 \geq t > q_4 \\ \{\} & \\ \begin{matrix} D \\ F \end{matrix} SIIprt, & t > q_5 \\ \ldots & \end{cases}$$

$\begin{matrix} C & A \\ DSIIprt & B \\ F & Q \end{matrix}$ is considered above. In particular, $\begin{matrix} Q & A \\ DSIIprt & B \\ F & Q \end{matrix}$ can be interpreted as a game: player 1 fits A into Q by B, and the other pushes F out of Q by D 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.

Can be considered discrete hierarchical structure, continuous hierarchical structure, and discrete-continuous hierarchical structure [3],
$Sprt_x^{\text{N–hierarchical structure}}$
The example

$$\text{QHSIIpr}=\text{H}SIIprt\left[\left[\begin{matrix}\text{N – level of hierarchical structure}\\ SIIprt\ B\\ x\\ \dots\\ \text{i – level of hierarchical structure}\\ SIIprt\ B\\ x\\ \dots\\ \text{1 – level of hierarchical structure}\\ SIIprt\ B\\ x\\ B\\ x\end{matrix}\right]\right]$$

-- N-hierarchical structure compression into point x by B.

Let f(N, QHSIIpr)= $\text{QHSIIpr}^{\text{QHSIIpr}^{\text{QHSIIpr}^{\cdots^{\text{QHSIIpr}}}}}\Big\}$-N levels

It can be considered selfII- QHSIIpr, f(y, QHSIIpr) for any y, f(QHSIIpr, QHSIIpr).
Compression Hierarchy Examples:

$$1)\ SIIprt\quad SIIprt(\ )\quad \begin{matrix}()\\ SIIprt(\ )\\ ()\\ ()\\ ()\\ SIIprt(\ )\\ ()\\ ()\\ SIIprt(\ )+B\\ ()\end{matrix}=\left(\begin{matrix}()\\ SIIprt(\ )\\ ()\\ ()\\ SIIprt\ SIIprt(\ )\\ ()\\ ()\\ SIIprt(\ )\\ ()\\ ()\\ SIIprt\ ()\\ SIIprt(\ )\\ ()\\ B\end{matrix}\right)$$

2) $\begin{matrix} C+{}^{()}_{()}Sprt^{()}_{()} \\ D+{}^{()}_{()}Sprt^{()}_{()} \end{matrix} S_1 t \begin{matrix} A+{}^{()}_{()}Sprt^{()}_{()} \\ B+{}^{()}_{()}Sprt^{()}_{()} \end{matrix} = \begin{pmatrix} {}^{()}_{()}Sprt^{()}_{()} & {}^{()}_{()}Sprt^{()}_{()} \\ {}^{()}_{()}Sprt^{()}_{()} S_1 t & {}^{()}_{()}Sprt^{()}_{()} \\ {}^{()}_{()}Sprt^{()}_{()} & {}^{()}_{()}Sprt^{()}_{()} \\ {}^{C}_{D} S_1 t^{A}_{B} & \end{pmatrix}$

$$\begin{matrix} C + SIIprt^{()}_{()} & A + SIIprt^{()}_{()} \\ R + SIIprt^{()}_{()} & SIIprt\, B + SIIprt^{()}_{()} \\ D + SIIprt^{()}_{()} & Q + SIIprt^{()}_{()} \end{matrix} = \begin{pmatrix} {}^{()}_{()} & {}^{()}_{()} \\ SIIprt^{()}_{()} & SIIprt^{()}_{()} \\ {}^{()}_{()} & {}^{()}_{()} \\ SIIprt^{()}_{()} & SIIprt\ SIIprt^{()}_{()} \\ {}^{()}_{()} & {}^{()}_{()} \\ {}^{()}_{()} & {}^{()}_{()} \\ SIIprt^{()}_{()} & SIIprt^{()}_{()} \\ {}^{()}_{()} & {}^{()}_{()} \\ {}^{C}_{R} & {}^{A}_{SIIprt\, B} \\ {}^{D} & {}^{Q} \end{pmatrix}$$

The example of variable hierarchy

$$\begin{matrix} C & A \\ RSIIprt(t) & B \\ F & Q \end{matrix} = \begin{cases} \left\{ \begin{matrix} D \\ Q \\ A \cup B \cup C - A \cap B \cup (A \cap B \cap C) \end{matrix} \right\}, , \quad q_2 \geq t \geq q_1 \\ {}^{A}_{SIIprt\, B}, q_3 \geq t > q_2 \\ {}^{C}_{RSIIprt}, q_4 \geq t > q_3 \\ {}^{A}_{ASIIprt(t)\, B}, q_5 \geq t > q_4 \\ {}^{C}_{RSIIprt(t)\, A}, \quad t > q_5 \\ \ldots \end{cases} \qquad (*_{6.2}),$$

where D is self[2]-set for $A \cap B \cap C$, Q is self-set for $A \cap B - A \cap B \cap C$.

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 selfII-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 selfII-object

$SIIprt(t)\begin{matrix}A_0+E_s\\A_0+E_s\end{matrix}$, where $SIIprt(t)\begin{matrix}A_0\\A_0\end{matrix}$ - level of objectivity of an amorphous object,

$$P = (SIIprt(t)\begin{matrix}A_0\\A_0+E_s\end{matrix} + SIIprt(t)\begin{matrix}A_0\\A_0\end{matrix} + SIIprt(t)\begin{matrix}A_0+E_s\\A_0\end{matrix} SIIprt(t)\begin{matrix}A_0\\A_0+E_s\end{matrix} + SIIprt(t)\begin{matrix}A_0+E_s\\A_0\end{matrix} + SIIprt(t)\begin{matrix}A_0+E_s\\A_0+E_s\end{matrix})$$ - the energy of connections between the level

of subtle energy $SIIprt(t)\begin{matrix}E_s\\E_s\end{matrix}$ 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

$$SIIprt(t)\begin{matrix}E_s\\E_s\\E_s\end{matrix}\left(\begin{matrix}\\P\\\end{matrix}\right)\begin{matrix}A_0\\A_0\\A_0\end{matrix} \quad (1.7.1).$$

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

$$\begin{matrix} A_0 + E_s \\ SIIprt(t)A_0 + E_s \\ A_0 + E_s \\ A_0 + E_s \\ SIIprt(t)SIIprt(t)A_0 + E_s \\ A_0 + E_s \\ A_0 + E_s \\ SIIprt(t)A_0 + E_s \\ A_0 + E_s \end{matrix} \quad (1.7.2).$$

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

$$\begin{matrix} \{\} & A_0 + E_s \\ SIIprt(t)\{\} & SIIprt(t)A_0 + E_s \\ \{\} & A_0 + E_s \\ \{\} & A_0 + E_s \\ SIIprt(t)\{\} & SIIprt(t)SIIprt(t)A_0 + E_s \\ \{\} & A_0 + E_s \\ \{\} & A_0 + E_s \\ SIIprt(t)\{\} & SIIprt(t)A_0 + E_s \\ \{\} & A_0 + E_s \end{matrix}$$

and energy, it behaves like a crystal: , a nutrient medium appears and the necessary energy:

$$\begin{matrix} B_0 + E_q & B_0 + E_q \\ SIIprt(t)B_0 + E_q & SIIprt(t)B_0 + E_q \\ B_0 + E_q & B_0 + E_q \\ B_0 + E_q & B_0 + E_q \\ SIIprt(t)B_0 + E_q & SIIprt(t)SIIprt(t)B_0 + E_q \\ B_0 + E_q & B_0 + E_q \\ B_0 + E_q & B_0 + E_q \\ SIIprt(t)B_0 + E_q & SIIprt(t)B_0 + E_q \\ B_0 + E_q & B_0 + E_q \end{matrix}$$

, its structure is transformed into a

$$B_0 + E_q \qquad A_0 + E_s + B_0 + E_q$$
$$SIIprt(t)B_0 + E_q \qquad SIIprt(t)A_0 + E_s + B_0 + E_q$$
$$B_0 + E_q \qquad A_0 + E_s + B_0 + E_q$$
$$B_0 + E_q \qquad A_0 + E_s + B_0 + E_q$$

mathematical structure $SIIprt(t)B_0 + E_q \quad SIIprt(t)SIIprt(t)A_0 + E_s + B_0 + E_q$. The

$$B_0 + E_q \qquad A_0 + E_s + B_0 + E_q$$
$$B_0 + E_q \qquad A_0 + E_s + B_0 + E_q$$
$$SIIprt(t)B_0 + E_q \qquad SIIprt(t)A_0 + E_s + B_0 + E_q$$
$$B_0 + E_q \qquad A_0 + E_s + B_0 + E_q$$

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:

$$\begin{array}{r} A_0 + E_s \\ SIIprt(t)A_0 + E_s \\ A_0 + E_s \\ A_0 + E_s \\ SIIprt(t)SIIprt(t)A_0 + E_s \\ A_0 + E_s \\ A_0 + E_s \\ SIIprt(t)A_0 + E_s \\ A_0 + E_s \\ A_0 + E_s \\ SIIprt(t)A_0 + E_s \\ A_0 + E_s \\ A_0 + E_s \\ SIIprt(t)SIIprt(t)A_0 + E_s \end{array} \quad \begin{array}{r} A_0 + E_s \\ SIIprt(t)A_0 + E_s \\ A_0 + E_s \\ A_0 + E_s \\ SIIprt(t)SIIprt(t)A_0 + E_s \\ A_0 + E_s \\ A_0 + E_s \\ SIIprt(t)A_0 + E_s \\ A_0 + E_s \\ A_0 + E_s \\ SIIprt(t)A_0 + E_s \\ A_0 + E_s \\ A_0 + E_s \\ SIIprt(t)SIIprt(t)A_0 + E_s \end{array}$$

. Next comes the level

$$\begin{array}{r} A_0 + E_s \\ A_0 + E_s \\ SIIprt(t)A_0 + E_s \\ A_0 + E_s \\ A_0 + E_s \\ SIIprt(t)A_0 + E_s \\ A_0 + E_s \\ A_0 + E_s \\ SIIprt(t)SIIprt(t)A_0 + E_s \\ A_0 + E_s \\ A_0 + E_s \\ SIIprt(t)A_0 + E_s \\ A_0 + E_s \end{array} \quad \begin{array}{r} A_0 + E_s \\ A_0 + E_s \\ SIIprt(t)A_0 + E_s \\ A_0 + E_s \\ A_0 + E_s \\ SIIprt(t)A_0 + E_s \\ A_0 + E_s \\ A_0 + E_s \\ SIIprt(t)SIIprt(t)A_0 + E_s \\ A_0 + E_s \\ A_0 + E_s \\ SIIprt(t)A_0 + E_s \\ A_0 + E_s \end{array}$$

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.

$A(t)$
$A(t)SIIprt$ will be called dynamic anti-capacity from oneselfII. 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 and the experiments in chemistry Nazhipa Valitov correspond to the concept of the Universe as a capacity in itselfII as the element. They experimented with connections for elements of the microworld, and since here the connections are selfII-connections, then when the object component of selfII-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 selfII-containment in itselfII is closed on itselfII.

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 selfII-objects: for example,1) ${}_{2}^{3}He$, ${}_{2}^{3}He$ and the formation of one selfII-object ${}_{2}^{4}He$, 2) ${}_{2}^{3}He$, ${}_{1}^{2}H$ and the formation of one selfII-object ${}_{2}^{4}He$. As a result, the energy of the collapse of the lost part of the selfII is released.

Remark 1.7.2. To gain access to object transformation, just go to the level lS= $\frac{2}{\pi}arctg(1+\mathcal{E})$, $\mathcal{E}$ may be quite small.

Examples of transformation:

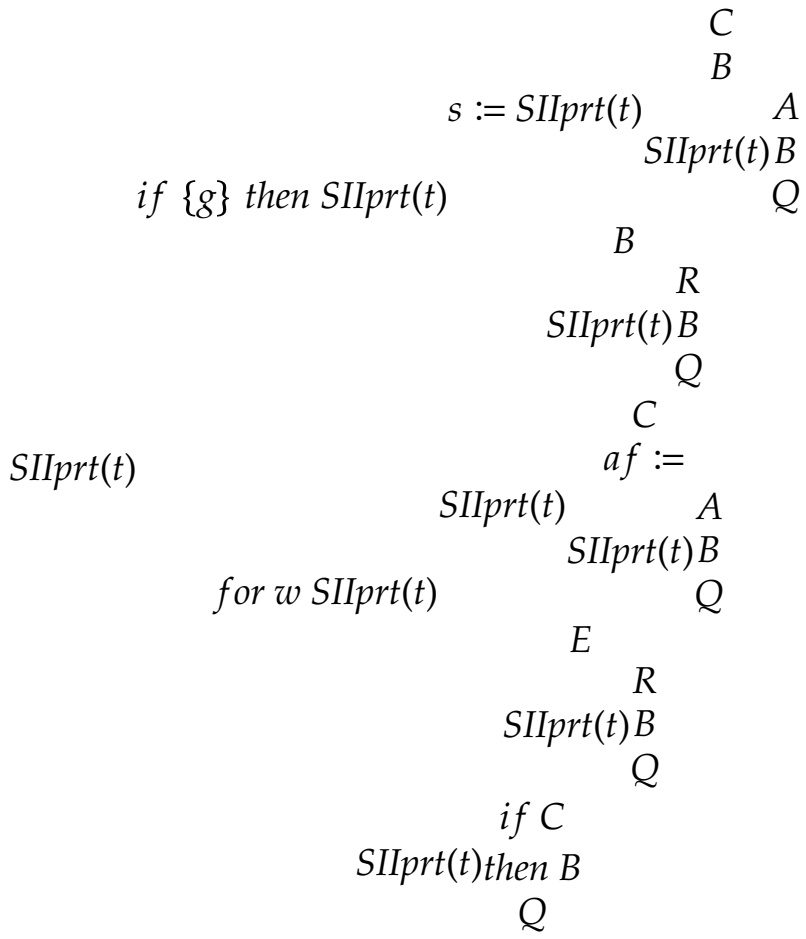
$$1)\ \begin{matrix} q \\ SIIprt(t)q \\ q \end{matrix} \rightarrow \begin{matrix} q \\ q \\ SIIprt(t)SIIprt(t)q \\ q \\ q \end{matrix} \rightarrow \begin{matrix} b \\ SIIprt(t)_c \\ d \\ b \\ SIIprt(t)SIIprt(t)_c \\ d \\ b \\ SIIprt(t)_c \\ d \end{matrix}$$

2) $\overset{q}{SIIprt(t)}\overset{q}{q} \rightarrow \mathrm{SII_3f(selfII(q))} \rightarrow \overset{r}{SIIprt(t)}\overset{r}{r}$

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.

SelfII A $=\overset{A}{SIIprt}\overset{A}{A}$ can be transformed into any D if μl(D) = μl ($Sprt_A^A$), μl(x) - level measure of selfII for x, in particular, into $SIIprt\ \overset{any\ C}{\underset{A}{A}}$ or $SIIprt\ \overset{A}{\underset{any\ C}{A}}$, and also an object R into any object Q or any energy U. The transformations of this type will be called stransformations. $\mathrm{SelfII^N}\overset{A}{A}$ can transform itselfII into any D if N ≥2; to realize this we need an even larger quantity N.

1) Example of a parallel-serial program statement



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

SelfII^{N}- OS and is designed for such transformations, and it itselfII can be transformed at N≥1, or it itselfII can be transformed at N ≥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 selfII-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 selfII-level of the process, the definition of selfII-values (selfII-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 selfII-level of chemical reactions, but here there will be a trivial identification and determination of the selfII-level will be much simpler. For example, a type w ≡ 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 1.7.5. Let us consider an analogue of the Schrödinger equation for networks operating on electromagnetic energy

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

w- measure of selfII: mS(Q), $\mu S(H)$ - measure of selfII for H, H = H(μS(p), μS (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 1.7.6. The selfII-space of a higher level contains many selfII-energetic fibers, collecting into appropriate sets that can be accessed by the corresponding selfII-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 selfII-space with objects.

Remark 1.7.7. It is quite possible to try to build up the levels of objects and processes; change something at these levels.

Remark 1.7.8. One can try to conventionally represent the mathematical model (1.7.1) of the atom (molecule) as a hierarchical dynamic operator.

Remark 1.7.9. Here, selfII-action is understood as action on oneselfII (i.e., to the same action), while physicists understand selfII-action, for example, as the absorption of one elementary particle by another of the same type.

Remark 1.7.10. A selfII-molecule (selfII-atom, oselfII-(elementary particle))) as a capacity can have the following types of selfII, oselfII: selfII-set, selfII-structure, selfII-hierarchy or its elements that generates this selfII-molecule (selfII-atom, oselfII-(elementary particle))). SelfII-power is force that is applied to oneselfII or its elements that generates this selfII-power.

Remark 1.7.11. Similarly, everything discussed above in this part can be transferred to cases with fuzzy containment and fuzzy objects.

1.8 PROGRAM OPERATORS SIIprt, tprSII, SII^1epr, $SIIeprt_1$

Here it is supposed to use a symbiosis of parallel actions and conventional calculations through sequential actions. This must be done through SIIprt-Networks - analogue of Sit-Networks [13] in one of the central departments of which a conventional computer system is located. The parallel processor is itself [13] with direct parallel computing not through serial computing.

Using conventional coding by a computer system, through a Target-block with a SIIprt -program operator - $\text{SIIprt}\ {}^{Ag}_{B}{}_{activation}$ 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 $\text{SIIprt}\ {}^{\{\text{UHF AC} := Q\}}_{B}{}_{activation}$ by B, 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 program operators:

1) SIIprt-program operators
2) tprSII-program operators
3) SII^{1}epr - program operators
4) SIIeprt$_{1}$- program operators,

   which can work with fuzzy arguments as well.

One example is pattern recognition: SIIprt

if SIIprt $\underset{B}{\overset{Q}{image\ archive}}$ ∋ SIIprt$\underset{B}{\overset{q}{\mu}}$ $\underset{B}{\overset{Q}{then\ Name\ of\ q}}$.

The example of SIIprt-program is SIIprt

{ SIIprt $\underset{x}{\overset{F}{\{\{p\} := \{a(x)\}\}}}$, SIIprt $\underset{x}{\overset{F}{IF\{\{B\}\{f\}\}\ then\ Q}}$, SIIprt $\underset{Q}{\overset{Q}{g}}$ } .

Consider a third type of capacity in itself - analogue of capacity in itself. For example, based on SIIprt$\underset{q}{\overset{\tilde{x}}{B}}$, where $\tilde{x} = (x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \ldots, x_n|\mu_{\tilde{x}}(x_n))$, 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 SIIprt$\underset{q}{\overset{\tilde{x}}{g}}$ contains only m elements. Form (1.1) can be generalized into the following forms: (1.1.1) – (1.4).

The ideology of SIIprt and SII$_3$f can be used for programming.

Consider a third type of capacity in itself. For example, based on $\text{SIIprt}^{\tilde{x}}_{q}g$, where $\tilde{x}$ $=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2)|,\ldots,x_n|\mu_{\tilde{x}}(x_n)|)$ i.e. n - elements at one point, we can consider the fuzzy capacity $SII_3 f\mu$ in itself with m elements from $\tilde{x}$, m<n, which is formed according to the form from forms (1.1) – (1.4).

Here are some of the SIIprt-program operators.

1. Simultaneous assignment by B of the expressions $\tilde{p}=(p_1|\mu_{\tilde{p}}(p_1), p_2|\mu_{\tilde{p}}(p_2)_{,\ldots,} p_n|\mu_{\tilde{p}}(p_n))$ to the variables $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2)_{,\ldots,} x_n|\mu_{\tilde{x}}(x_n))$. This is implemented via $\text{SIIprt}^{\{\{\tilde{x}\}:=\{p\}\}}_{B\ w}$.
2. Simultaneous checking by $\tilde{d}$ by the fuzzy set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2)_{,\ldots,} g_n|\mu_{\tilde{g}}(g_n))$ for the fuzzy set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2)_{,\ldots,} B_n|\mu_{\tilde{B}}(B_n))$ Implemented via $\text{SCprt}^{IF\{\{\tilde{B}\}\{\tilde{g}\}\}\ then\ \tilde{Q}}_{\tilde{d}\ w}$ where $\tilde{Q}$ can be anything.
3. Similarly for fuzzy loop operators and others.

$SII_3 f$–software operators will differ only just because aggregates $\tilde{x},\tilde{p}, \tilde{B},\tilde{d}$ will be formed from corresponding SIIprt-program operators in form (1.1), for more complex operators in forms (1.1.1) - (1.4).

For example, $\text{SIIprt}^{\{R\}}_{B\ w\{R\}}$ is the fuzzy capacity by B in itself of the second type if $w\{R\}$ is a program capable of generating $\{R\}$ by B.

The example of self-ffprogram of the first type is

$$\text{SIIprt}\{\ \text{SIIprt}^{\{\{p\}:=\{\mathsf{a}(x)\}\}}_{B\ w}\ ,\ \text{SIIprt}^{IF\{\{B\}\{f\}\}\ then\ Q}_{B\ w}\ ,\ \text{SIIprt}^{Q}_{B\ w}g\ \}.$$

The example of SIIprt-program for SmnSIIprt:

$\underset{B}{\overset{q}{\underset{d}{:=}}}$ SCprt - fuzzy assigning fuzzy q to fuzzy B by d.

$\underset{q}{\overset{tw}{\underset{d}{:=}}}$ SCprt - assigning target weight *tw* to q by d.

$\underset{d}{\overset{\{q\}w}{}}$ SIIprt - SmnSIIprt *activation* for fuzzy $\{q\}w$ by d.
SmnSIIprt *activation*

SIIprt-coding

SIIprt-coding by d: 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, $\underset{B}{\overset{A}{\underset{d}{\text{SCprt}}}}$.

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

SIIelf-coding

ffSelf-coding by d: 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, $\underset{A}{\overset{A}{\underset{d}{\text{SIIprt}}}}$.

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 SIIprt-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 ffSIIprt -connectors in SIIprt - coding, for example: SIIprt $\underset{activation}{\{\text{UHF AC} \underset{d}{:=} Q\}}$. UHF AC field activation is used.

Dynamic SIIprt and $SII_3f(t)$ programming

The ideology of dynamic SIIprt and $SII_3f(t)$ can be used for programming:

1. Simultaneous assignment of the expressions $\widetilde{p(t)}$=($p_1(t)|\mu_{\widetilde{p(t)}}(p_1(t))$, $p_2(t)|\mu_{\widetilde{p(t)}}(p_2(t))$, ..., $p_n(t)|\mu_{\widetilde{p(t)}}(p_n(t))$) 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 $\underset{w(t)}{\overset{d}{\left\{\widetilde{x(t)}\right\}:=\left\{p(t)\right\}}}$ SIIprt(t).
2. Simultaneous checking the fuzzy set of conditions $\widetilde{g(t)}$=($g_1(t)|\mu_{\widetilde{g(t)}}(g_1(t))$, $g_2(t)|\mu_{\widetilde{g}(t)}(g_2(t))$, ..., $g_n(t)|\mu_{\widetilde{g(t)}}(g_n(t))$) for the fuzzy set of expressions $\widetilde{B(t)}$=($B_1(t)|\mu_{\widetilde{B(t)}}(B_1(t))$, $B_2(t)|\mu_{\tilde{B}(t)}(B_2(t))$, ..., $B_n(t)|\mu_{\widetilde{B(t)}}(B_n(t))$). Implemented via SIIprt(t) $\underset{w(t)}{\overset{d}{IF\left\{\left\{\widetilde{B(t)}\right\}\left\{\tilde{g}(t)\right\}\right\}\ then\ \widetilde{Q(t)}}}$ where $\widetilde{Q(t)}$ can be anything.
3. Similarly for fuzzy loop operators and others.

$SII_3f(t)$– 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 SIIprt-program operators in form (1.1) for more complex operators in forms (1.1.1) - (1.4)

tprSII-program operators

The ideology of tprSII and t_{SC_4f} - analogues of tS and t_{S_4f} from [13] can be used for programming. Here are some of the tprSC -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 $\tilde{x}$=($x_1|\mu_{\tilde{x}}(x_1)$, $x_2|\mu_{\tilde{x}}(x_2)$, ..., $x_n|\mu_{\tilde{x}}(x_n)$). This is implemented via $\underset{=:\tilde{p}}{\overset{\tilde{x}}{d}}$ SIIprt.
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 $\underset{IF\{\{\tilde{B}\}\{\tilde{g}\}\} \ then \ \tilde{Q}}{\overset{w}{\overset{d}{}}}$ SIIprt where Q can be any.

3. Similarly for loop operators and others.

t_{SII_4f} –software operators will differ only just because aggregates $\tilde{x}$,$\tilde{p}$, $\tilde{B}$,$\tilde{g}$ will be formed from corresponding tprSII program operators in form (1.1) for more complex operators in forms (1.1.1) - (1.4).

Consider hierarchical tprSII-program operator

$\underset{A}{\overset{B}{d}}$SIIprt $= \left\{ \begin{matrix} D + \underset{}{\overset{\{\}}{\overset{d}{}}} \text{SIIprt} \\ A - A \cap B \\ (B - A \cap B) \end{matrix} \right\}$, where D is oself-(fuzzy set) for fuzzy $(A \cap B)$.

Dynamic tprSII and $t(q)_{SII_4f}$ programming at time q

The ideology of tprSII and t_{SII_4f} can be used for dynamic programming. Here are some of the tprSII(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 $\widetilde{x(t)}$=$(x_1(t)|\mu_{\widetilde{x(t)}}(x_1(t)), x_2(t)|\mu_{\tilde{x}(t)}(x_2(t))_{, ..., } x_n(t)|\mu_{\widetilde{x(t)}}(x_n(t)))$.is implemented through $\underset{= :\tilde{p}(t)}{\overset{\tilde{x}(t)}{\overset{d}{}}}$ SIIprt(t).

2. The process of simultaneous expelling check the fuzzy set of conditions $\widetilde{g(t)}$ =$(g_1(t)|\mu_{\widetilde{g(t)}}(g_1(t)), g_2(t)|\mu_{\tilde{g}(t)}(g_2(t))_{, ..., } g_n(t)|\mu_{\widetilde{g(t)}}(g_n(t)))$ for the fuzzy set of expressions $\widetilde{B(t)}$=$(B_1(t)|\mu_{\widetilde{B(t)}}(B_1(t)), B_2(t)|\mu_{\tilde{B}(t)}(B_2(t))_{, ..., } B_n(t)|\mu_{\widetilde{B(t)}}(B_n(t)))$ is implemented through $\underset{IF\{\{\tilde{B}(t)\}\{\tilde{g}(t)\}\} \ then \ \tilde{Q}(t)}{\overset{w(t)}{\overset{d}{}}}$ SIIprt(t), where Q(t) can be any.

3.Similarly for loop operators and others.

t_{SC_4f} –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 tprSII(t) for above mentioned programming operators through form (1.1) or form (1.1.1) - (4) for more complex operators.

Consider hierarchical dynamic tprSII-program operator:

$$\overset{B(q)}{\underset{A(q)}{g}}\text{SIIprt(q)} = \left\{ \begin{array}{c} t(q)_{SC_1 f(A(q)\cap B(q))} + \overset{\{\}}{g}\text{SIIprt(q)} \\ A(q) - A(q)\cap B(q) \\ \left(B(q) - A(q)\cap B(q)\right) \end{array} \right\},$$

SII[1]epr -program operators (form $\underset{D}{\overset{B}{g_2}}\text{SII}^1\text{prt}\underset{B}{\overset{A}{g_1}}$ - analogue of ${}_D^B S^1 t_B^A$ [3]**))**

For example, $\underset{\{=:\{p(t)\}\}}{\overset{\{\mathsf{a}(t)\}}{g_2}}\text{SII}^1\text{prt}\underset{\{\mathsf{a}(t)\}}{g_1}$ $IF\{\{B(t)\}\{f(t)\}\}$ *then* $Q(t)$.

Consider hierarchical dynamic SII[1]epr-program operator: (form $\begin{matrix} \overset{B}{g_2}\text{SII}^1\text{prt}\overset{A}{g_1} * \\ \overset{A}{B}\quad\overset{B}{A} \\ \underset{D-A}{g_2}\ \text{SII}^1\text{prt}\underset{B}{g_1} \end{matrix}$).

SIIeprt$_1$- program operators (form $\underset{D}{\overset{C}{g_2}}\text{SII}_1\text{prt}\underset{B}{\overset{A}{g_1}}$ - analogue of ${}_D^C S_1 t_B^A$ [8]**))**

${}_a^a f S_1 t_a^a$-- sample $\begin{pmatrix} self \\ oself \end{pmatrix}$-fprogram structure example.

Consider structure examples hierarchical fuzzy Set$_1$-program operator

4. $\begin{pmatrix} S_{01}^{et} fB \\ {}_{R-B}^{Q-B} S_1 t_B^{A-B} \end{pmatrix}$,
5. $\begin{pmatrix} S_{21}^{et} f_A^B \\ {}_{R-A}^{Q-A} S_1 t_B^A \end{pmatrix}$,
6. $\text{SIIprt}\left\{ \begin{array}{c} {}^{q}_{W_q}\left(\begin{smallmatrix} a & a \\ g_1\text{SC1rt}g_1 \\ a & a \end{smallmatrix}\right) St_{q}^{E_q} \\ \overset{g}{t_0} \end{array}, \quad St_{d_r}^{\{El^{d_r}\}} \right\}$ —program structure example, where

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

7. $\underset{\substack{g \\ t_0}}{\text{SIiprt}}\left\{\begin{matrix} {}^{q}\begin{pmatrix} a & a \\ g_1\text{SC1rt}g_1 & \\ a & a \end{pmatrix} St_{q}^{E_q} \\ W_q \end{matrix}, \quad St_{d_r}^{\left\{El^{d_r}\right\}}\right\}$ can be interpreted as a program operator.

$\begin{pmatrix} a & a \\ g_1\text{SII}^1\text{rt}g_1 & \\ a & a \end{pmatrix}$ can be interpreted as a $\begin{pmatrix} s^1elf \\ os^1elf \end{pmatrix}$-fprogram operator. $\begin{pmatrix} a & a \\ g_1\text{SII}_1\text{rt}g_1 & \\ a & a \end{pmatrix}$ sample $\begin{pmatrix} s_1elf \\ os_1elf \end{pmatrix}$-fprogram structure example.

Appendix

Remark. Energy of a living organism:

$$\text{Cg(r, a}(E_q)) = \underset{\substack{g \\ t_0}}{\text{SCprt}}\left\{\begin{matrix} {}^{q}\begin{pmatrix} a & a \\ g_1\text{SIIrt}g_1 & \\ a & a \end{pmatrix} fSt_{q}^{E_q} \\ W_q \end{matrix}, \quad fSt_{d_r}^{\left\{El^{d_r}\right\}}\right\}$$

$\begin{pmatrix} a & a \\ g_1\text{SIIrt}g_1 & \\ a & a \end{pmatrix}$-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^{d_r} - a bundle of fibers of external energy self-capacities, collected at the point of assembly of the cocoon of a living organism.

Entire neural network as instantaneous simultaneous CRAM in SIIprt-elements. When activated in a neural network, the entire neural network becomes a working memory. Use of self-energy as fuzzy activation or from outside. CQ_0= SIIprt

$\text{SIIprt}\ {}^{S_{mn\text{SII}prt}}_{g\ activation} \rightarrow$ self-CRAM, $CIIQ_{00}=\ {}^{S_{mn\text{SII}prt}}_{g\ activation}\ \text{SIIprt}$ $CIIQ_0$prt, $CIIQ_{01}=\ \text{SIIprt}\ {}^{S_{mn\text{SII}prt}}_{g\ activation}$

$\text{SIIprt}\ {}^{S_{mn\text{SII}prt}}_{g\ activation}$ $CIIQ_0$.$CIIQ_0$,$CIIQ_{00}$,$CIIQ_{01}$-coding,translation,realization in $\text{SIIprt}\ {}^{S_{mn\text{SII}prt}}_{g\ activation}$

eprograms, use $CIIQ_0$,$CIIQ_{00}$,$CIIQ_{01}$-$S_{mn\text{SII}prt}$,CIIAssembler.

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: SIIprt-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
4) Ershov Y.L. On a Hierarchy of Sets I, Albebra and Logic, 7 (1968), 47 - 73.
5) Ershov Y.L.] On a Hierarchy of Sets II, Albebra 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. Sptinger-Verllag, 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 «ΛΌΓΟΣ» 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) 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 https://dynamical-math.com/materials.html

14) Illia Danilishyn, Oleksandr Danilishyn. Introduction to Dynamic Mathematics: dynamic sets, dynamic operators and their applications: monograph / ed. by Pasynkov V. – Shawnee, USA : Primedia eLaunch LLC. – 2024. – 442 p. ISBN 979-8-89217-806-8

15) 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

16) 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. Introduction to Dynamic Operators: Lprt-Elements and Applications to Physics and Other their Applications. https://www.opastpublishers.com/journal/journal-of-mathematical-techniques-and-computational-mathematics/articles-in-press

17) Illia Danilishyn, Oleksandr Danilishyn. Introduction to Dynamic Mathematics: Zprt-elements and Their Applications. American Journal of Mathematical and Computer Applications, Volume 1 | Issue 1 (2025), pp.1-146. https://cskscientificpress.com/articles_file/278-_article1742642034.pdf

18) Illia Danilishyn, Oleksandr Danilishyn. Mathematical fundamentals of constructing pseudoliving energy theory and dynamic programmings: monograph / ed. by Pasynkov V. – Shawnee, USA: Primedia eLaunch LLC. – 2025. – 382 p. ISBN 979-8-89660-275-0 DOI 10.36074/mfocp-letadp.monograph-2025. https://dynamical-math.com/materials.html

19) Illia Danilishyn, Oleksandr Danilishyn. Mathematical uncertainties and their applications: monograph / ed. by Pasynkov V. – Shawnee, USA : Primedia

eLaunch LLC. – 2025. – 612 p. ISBN 979-8-89660-284-2. DOI: https://doi.org/10.36074/muata.monograph-2025 https://dynamical-math.com/materials.html

20) Illia Danilishyn and Oleksandr Danilishyn. Elements of Dynamic Science. Journal of Modern Classical Physics & Quantum Neuroscience. Vol:1,3. Pg:1-35. DOI: doi.org/10.63721/25JPQN0112. https://wmjournals.com/physics.html

21) I. Danilishyn, & O. Danilishyn, (2025). Elements of dynamic science: monograph. *Primedia ELaunch LLC*, 573. https://doi.org/10.36074/eods.monograph-2025

22) 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

23) Danilishyn I.V. Danilishyn O.V. THE USAGE OF SIT-ELEMENTS FOR NETWORKS. IV International Scientific and Practical Conference ”GRUNDLAGEN DER MODERNEN WISSENCHAFTLICHEN FORSCHUNG ”, 31.03.2023/Zurich, Switzerland. https://archive.logos-science.com/index.php/conference-proceedings/issue/view/9

24) Danilishyn I.V. Danilishyn O.V. DYNAMICAL SIT-ELEMENTS. IV International Scientific and Practical Conference ”GRUNDLAGEN DER MODERNEN WISSENCHAFTLICHEN FORSCHUNG ”, 31.03.2023/Zurich, Switzerland. https://archive.logos-science.com/index.php/conference-proceedings/issue/view/9

25) 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 «ΛΌΓΟΣ» 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**

26) 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**

27) 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

Dynamic Sets Theory: Sit-elements and Their Applications | Research Square

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

28) Danilishyn I., Danilishyn O. VARIABLE HIERARCHICAL DYNAMICAL STRUCTURES (MODELS) FOR DYNAMIC, SINGULAR, HIERARCHICAL SETS AND THE PROBLEM OF COLD THERMONUCLEAR FUSIONThe 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**

Declarations:

Availability of data and material

1) Oleksandr Danilishyn, Illia Danilishyn. Introduction to Dynamic Sets Theory: SIIprt-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

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 WISSENCHAFTLICHEN 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 «ΛΌΓΟΣ» 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 WISSENCHAFTLICHEN 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 «ΛΌΓΟΣ» 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

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

13) Illia Danilishyn, Oleksandr Danilishyn. Introduction to Dynamic Mathematics: dynamic sets, dynamic operators and their applications: monograph / ed. by Pasynkov V. – Shawnee, USA : Primedia eLaunch LLC. – 2024. – 442 p. ISBN 979-8-89217-806-8

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

15) 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. Introduction to Dynamic Operators: Lprt-Elements and Applications to Physics and Other their Applications. https://www.opastpublishers.com/journal/journal-of-mathematical-techniques-and-computational-mathematics/articles-in-press

16) Illia Danilishyn, Oleksandr Danilishyn. Introduction to Dynamic Mathematics: Zprt-elements and Their Applications. American Journal of Mathematical and Computer Applications, Volume 1 | Issue 1 (2025), pp.1-146. https://cskscientificpress.com/articles_file/278-_article1742642034.pdf

17) Illia Danilishyn, Oleksandr Danilishyn. Mathematical fundamentals of constructing pseudoliving energy theory and dynamic programmings: monograph / ed. by Pasynkov V. – Shawnee, USA: Primedia eLaunch LLC. – 2025. – 382 p. ISBN 979-8-89660-275-0 DOI 10.36074/mfocp-letadp.monograph-2025. https://dynamical-math.com/materials.html

18) Illia Danilishyn, Oleksandr Danilishyn. Mathematical uncertainties and their applications: monograph / ed. by Pasynkov V. – Shawnee, USA : Primedia eLaunch LLC. – 2025. – 612 p. ISBN 979-8-89660-284-2. DOI: https://doi.org/10.36074/muata.monograph-2025 https://dynamical-math.com/materials.html

19) Illia Danilishyn and Oleksandr Danilishyn. Elements of Dynamic Science. Journal of Modern Classical Physics & Quantum Neuroscience. Vol:1,3. Pg:1-35. DOI: doi.org/10.63721/25JPQN0112. https://wmjournals.com/physics.html

20) I. Danilishyn, & O. Danilishyn, (2025). Elements of dynamic science: monograph. *Primedia ELaunch LLC*, 573. https://doi.org/10.36074/eods.monograph-2025

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. Mathematical fundamentals of interpretations theory (future science)

Introduction

Since any knowledge is an interpretation, it becomes necessary to construct the mathematical fundamentals of interpretation theory. Elements of such mathematics are given here. Let's associate classical Mathematics (and in general classical science, which we will call 2-interpretation) with the Numerical Form (1, 1), (because it is a consistent approach, consistent logic), which corresponds to the consistent nature of the construction and use of mathematics (science) and, in particular, mathematical logic. Then Dynamic Mathematics will be associated the Numerical form (2, 1), which corresponds to the ||| [18 -20] and the Numerical form (1, 2), which corresponds to the $|||^{-1}$[18 -20], the Numerical form (1, (2, 1)), which corresponds to the self, etc. (1, 2) means expansion: $|||^{-1}$ (disidentification of an object into two), (2, 1) – compression (identification (|||) of two objects into one). In fact, self-type (for example, self, oself) and |||-type are manifestations (projections) of 3-interpretation. $|||^{-1}$is the element, which is not an element of anyone. Conventional science approaches things through matter, meaning that everything, including energy (which must be interpreted through material carriers—particles), is interpreted through matter. In dynamic science, which we propose as an alternative, everything, including matter, is interpreted through energy. Matter is interpreted as energy closed in on itself (energy within itself). If this were not the case, energy would dissipate over time and matter would not exist.

The true bearers of this approach are true sorcerers, but they reject classifications. We use them, but only through one position of the assemblage point on people's cocoons, corresponding to the perception of our world. This is significantly less than the perception of true sorcerers, who can utilize a significantly larger number of assemblage point positions. Our world is an interpretation of the intersection of two great bands of emanations: the band of organic beings and the band of inanimate material objects, which is what prompts conventional science's approach

through matter. Dynamic science takes an approach through energy, starting from this same intersection, removing the axiom of regularity from the foundation of conventional science—set theory—just as Lobachevsky, by removing the axiom of parallelism from Euclidean geometry, led to the creation of Einstein's general theory of relativity. By removing the axiom of regularity, we gain access to unlimited possibilities for interpreting energy. Dynamic science emphasizes the uniqueness of the object. In conventional science, their characteristics are manipulated, which also belong to the mental self-nature of the people manipulating them. Exit into upper level by SmnSprt through ||| with tw, back through $|||^{-1}$ with tw. By changing spaces, places within spaces, objects, actions, etc., through |||, $|||^{-1}$ anything is possible. Perception of human A for object B: by A|||B, the next (A|||B) |||C = (A|||C) |||B then $|||^{-1}$(A|||C) |||B and receiving A|||C. Either we take from emptiness by $|||^{-1}$ or through substitution by |||. First attention leads to usual science, symbiosis of it with interpretations of second attention leads to dynamic science. In the first attention, a person's spacesuit in this world is perceived as their physical body; in the second attention, a person's spacesuit in the energy space is perceived as their energy body—a cocoon of emanations; and finally, in the third attention, a person has developed to such a degree that they no longer need a spacesuit. The physical bodies of objects are only interpretations of their self-nature. We must preserve these "spacesuits" of ours for the development and preservation of our consciousness to enter the third attention when we no longer need these "spacesuits".

2.1 Elements of mathematical theory of interpretations

2.1.1 Simplest operations with interpretations forms by containment

Here are considered simplest operations with simplest interpretations forms by containment for interpretations the same thing in the Universe.

Let’s consider the following interpretation forms of type (A, B)(C): interpretation (A, B) for C. For example, (A, B)(C) = (1, (2, 1))(C) = self(C), (1, (3/2, 1))(C) is 3/4 part of self for C, (A, B)(C) = (2, (1, 2))(C) = oself(C) etc.

We denote the addition sign for interpretation forms by $+_{in}$. Addition operation with interpretation forms:

$(A, B)(C) +_{in} (Q, R)(C) = (A + Q, B + R)(C)$,

A, Q are homogeneous any, B, R are homogeneous any.

Remark 2.1.1.1. In particular, A and Q can be other structurally homogeneous forms. The same applies to B and R.

For example, $(2, 1)(C) +_{in} (1, 2)(C) =(3, 3)(C)$,

$(1, (2, 1))(C) +_{in} (1, (3, 2))(C) = (2, (5, 3))(C)$.

The following relationships hold:

1) Commutative law of addition
   $(A, B)(C) +_{in} (Q, R)(C) = (Q, R)(C) +_{in} (A, B)(C)$,
   $(A, B)(C) +_{in} (A, B)(D) = (A, B)(D) +_{in} (A, B)(C)$
   $(A, B)(C) +_{in} (Q, R)(D) = (Q, R)(D) +_{in} (A, B)(C)$
2) Associative law of addition
   $((A, B)(C) +_{in} (Q, R)(V)) +_{in} (C, D)(P) = (A, B)(C) +_{in} ((Q, R)(V) +_{in} (C, D)(P))$.

We denote the inverse operation sign for interpretation forms by $-_{in}$.

The subtraction operation with interpretation forms:

$(A, B) -_{in} (Q, R) = (A - Q, B - R)$.

We denote the multiplication sign for interpretation forms by $*_{in}$.

Multiplication operation with interpretation forms:

$(A, B) *_{in} (Q, R) = (A * Q, B * R)$

A, Q are homogeneous any, B, R are homogeneous any.

Remark 2.1.2. In particular, A and Q can be other structurally homogeneous forms. The same applies to B and R.

For example, $(2, 1) *_{in} (1, 2) =(2, 2)$,

$(1, (2, 1)) *_{in} (1, (3, 2)) = (1, (6, 2))$.

The following relationships are held:

1) Commutative law of multiplication
   $(A, B)(C) *_{in} (Q, R)(C) = (Q, R)(C) *_{in} (A, B)(C)$,
   $(A, B)(C) *_{in} (A, B)(D) = (A, B)(D) *_{in} (A, B)(C)$
   $(A, B)(C) *_{in} (Q, R)(D) = (Q, R)(D) *_{in} (A, B)(C)$
2) Associative law of multiplication
   $((A, B) *_{in} (Q, R)) *_{in} (C, D) = (A, B) *_{in} ((Q, R) *_{in} (C, D))$
3) Distributive law of multiplication with respect to addition
   $((A, B)(C) +_{in} (Q, R)(V)) *_{in} (C, D)(P) = (A, B)(C) *_{in} ((Q, R)(V) +_{in} (C, D)(P))$.
4) Distributive law of multiplication with respect to subtraction
   $((A, B)(C) -_{in} (Q, R)(V)) *_{in} (C, D)(P) = (A, B)(C) *_{in} ((Q, R)(V) -_{in} (C, D)(P))$.
5) $(A, B) *_{in} (1, 1) = (A, B)$
   From $(A, B) +_{in} (Q, R) = (C, D) +_{in} (Q, R)$ follows $(A, B) = (C, D)$
6) From $(A, B) *_{in} (Q, R) = (C, D) *_{in} (Q, R)$ follows $(A, B) = (C, D)$.

We denote the inverse operation sign for interpretation forms by $/_{in}$.

$(A, B) /_{in} (Q, R)$ or $\frac{(A, B)}{(Q, R)}_{in}$, $Q \neq 0, R \neq 0$.

The division operation with interpretation forms:

$(A, B) /_{in} (Q, R) = (A / Q, B / R) = (\frac{A}{Q}_{in}, \frac{B}{R}_{in})$, $Q \neq 0, R \neq 0$.

The fundamental property of a fraction

$$\frac{(A, B)*_{in}(C, D)}{(Q, R)*_{in}(C, D)}_{in} = \frac{(A, B)}{(Q, R)}_{in}, \quad Q \neq 0, R \neq 0$$

Operations with fractions

$$\frac{(A, B)}{(Q, R)}_{in} +_{in} \frac{(C, D)}{(Q, R)}_{in} = \frac{(A, B)+_{in}(C, D)}{(Q, R)}_{in}, \quad Q \neq 0, R \neq 0$$

$$\frac{(A, B)}{(Q, R)}_{in} +_{in} \frac{(C, D)}{(S, P)}_{in} = \frac{(A, B)*_{in}(S, P)+_{in}(C, D)*_{in}(Q, R)}{(Q, R)*_{in}(S, P)}_{in}, \quad Q \neq 0, R \neq 0, S \neq 0, P \neq 0,$$

$$\frac{(A, B)}{(Q, R)}_{in} *_{in} \frac{(C, D)}{(S, P)}_{in} = \frac{(A, B)*_{in}(C, D)}{(Q, R)*_{in}(S, P)}_{in},$$

$$\frac{(A, B)}{(Q, R)}_{in} /_{in} \frac{(C, D)}{(S, P)}_{in} = \frac{(A, B)*_{in}(S, P)}{(Q, R)*_{in}(C, D)}_{in},$$

$$\left((A, B) /_{in} (Q, R)\right)^n = \frac{(A, B)^n}{(Q, R)^{n\text{in}}}$$

The following relationship is held:

$$\sqrt{(A, B)} = (\sqrt{A}, \sqrt{B})$$

For example, $\sqrt{(4, 1)} = (\sqrt{2}, \sqrt{1})$.

Remark 2.1.1.3. Here is possible to use interpretation forms in the kind of algebraic groups, rings with one, fields and to create corresponding theory etc.

Partially 2-interpretation from 3-interpretation is in the form of (2, 1) and other notations ||| [17 -20]. For example,

A|||Q (*),

where, in particular, A and Q can be interpretation forms.

For example, one can obtain an interpretation of A in the form of Q using (*), or even replace A with Q.

Remark 2.1.1.4. Here is possible to use self-type interpretation forms, |||-type interpretation forms, games-forms and to create corresponding theories etc.

Remark 2.1.1.5. May consider interpretation forms by Q, where Q may be any not necessarily containment. May consider, for example, more general interpretation form ψ(A, g(B, C)) than scalar form (1, (2, 1)) and where elements connected by another action than containment or, for example, by structural containment into structure C via structure B. May consider more general interpretation form R(A, B), interpretation vector- forms, interpretation matrix- forms, interpretation tensor-forms, interpretation operator- forms etc.

Remark 2.1.1.6. May consider interpretation programming forms, form-programming.

Remark 2.1.1.7. May consider pa-interpretations, pa-classifications.

Also, we may consider interpretation forms (A, B) by classical vector algebra:

$\overline{a}$ = (A, B), $\overline{b}$ = (C, D).

1) Condition of collinearity (parallelism) of vectors: $\overline{a}$ and $\overline{b}$: $\overline{b} = \lambda \overline{a}$, где $\lambda \neq 0$.
2) vector lengths $\| \overline{a} \| = \sqrt{A^2 + B^2}$
3) the scalar product ($\overline{a}$, $\overline{b}$) = AC +BD
4) transposition $(2, 1)^T = \begin{pmatrix} 2 \\ 1 \end{pmatrix}$ – hierarchical |||
5) multiplication of matrix interpretation and vector interpretation $\begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} q \\ z \end{pmatrix} = \begin{pmatrix} aq \\ bz \\ cq \\ dz \end{pmatrix}$
6) etc.

For interpretation forms $\overline{a}$(t) = (A(t), B(t)) we may use differential and integral calculus, functional analysis, operator analysis, differential equations etc.

For example, $\frac{d(A(t), B(t))}{dx} = (\frac{dA(t)}{dt}, \frac{dB(t)}{dt}), (\frac{dA(t)}{dt}, \frac{\partial^2 B(x, t)}{\partial x \partial t})$.

Remark 2.1.1.7. 1.May consider any nonlinear interpretations, for example,

F((Q, R)(C)), sin(1, (x, 1)), g((2, 1), (1, 2)), g((x, 1), (1, y)), in Q(x(y)) by x(y) we will mean the substitution of y into the second argument in x, etc.

May consider any operators: self-operator is (1, (2, 1))-operator, which, in particular, has itself by own object of application, g((x, 1), (1, y))-operator, sin(1, (x, 1))-operator etc.

Further expansions of interpretations are possible, for example, as follows:

$(1, |||_f)$ is $(1, (2, 1))_)$- interpretation by f,

Self 2 times itself into one at the same time,

A|||A = selfA, ||| = A^{-1}selfA A^{-1},

A|||B = **(**A|||A)(A^{-1}|||B).

May input interpretation (G, V), interpretation $\begin{pmatrix} Q \\ C \end{pmatrix}$, interpretation $\begin{pmatrix} R & & B \\ & S & \end{pmatrix}$, interpretation $\begin{pmatrix} 1 & & \\ & 1 & \\ & & 1 \end{pmatrix}$. May input interpretation (1, 2, 1), where interpretation (1, 2) and interpretation (2, 1) are simultaneously, interpretation (2, 1, 2), where interpretation (2, 1) and interpretation (1, 2) are simultaneously, interpretation $\overrightarrow{(1, 2, 1)}$, where interpretation (2, 1) executes after interpretation (1, 2), interpretation $\overrightarrow{(2, 1, 2)}$, where interpretation (1, 2) executes after interpretation (2, 1), interpretation.

2.1.2 Syntax of self-like formations

The usual self(D) = D|||D and corresponds to the next higher level of the hierarchy and the form (1, (2, 1)).

General definition of self-type:

A $\overset{Z}{\rightarrow}$ itself, Z is any

If in the relation defining the automorphism we replace the "=" sign with the "≡" sign, we obtain the corresponding form of self.

Let's introduce the following concept of dself(A), A = $A_1 \cup A_2$, which consist from elements of some hierarchical levels: by self-level corresponds to element *a* from A_1, which is capable of generating A, and by |||-level corresponds to A_2 = Q*self(P(|||)), which has the embryo itself in the form of a compression of itself: R*paself(S(pa|||)) consists from all positions of the assemblage point and the

compression of the assemblage point itself into one point and is equal $\overset{w}{\mathrm{Q}*\mathrm{self}(\mathrm{P}(|||))}\mathit{Sprt}\overset{\mathrm{Q}*\mathrm{self}(\mathrm{P}(|||))}{w}$. Then for Vprt – element of a human A [20]

$$\left(\begin{array}{c} \overset{...}{} \\ parel f A\left(decignation - \overset{\equiv\equiv\equiv}{A}\right) \\ singel f A\left(decignation - \overset{\equiv\equiv\equiv}{A}\right) \\ subtle\ energy\ of\ object\ \ A\ \text{paradoxical}\ upper\ level\ (\ pa|||)\left(decignation - \overset{\equiv\equiv\equiv}{A}\right) \\ subtle\ energy\ of\ object\ \ A\ \text{paradoxical}\ mid - level(pasel f(A))\left(decignation - \overset{\equiv\equiv\equiv}{A}\right) \\ subtle\ energy\ of\ object\ \ A\ \text{paradoxical}\ down - level(psel f(A))\left(decignation - \overline{\equiv\equiv\equiv A}\right) \\ \begin{array}{cc} / & \backslash \\ subtle\ energy\ of\ |||^{-1} & subtle\ energy\ of\ ||| \\ \underline{\underline{\underline{A}}} & \overline{\overline{\overline{A}}} \\ \left(\underline{\underline{\underline{A}}}\right) & \left(\overline{\overline{\overline{A}}}\right) \end{array} \\ ordinary\ energy\ exhibited\ by\ an\ \ object\ \ A\left(decignation - \underline{\underline{\underline{A}}}\right) \leftarrow \ the\ raw\ energy\ of\ an\ object\ \ A\left(decignation - \overline{\overline{\overline{A}}}\right) \end{array}\right) \quad (**)$$

$\underline{\underline{\underline{A}}} = a$, $\overline{\overline{\overline{A}}} = \mathrm{Q}*\mathrm{self}(\mathrm{P}|||)$.

Example of self by form (1, (2, 1)): self(atom P) = $x_1|||_P x_2$, where x_1, x_2 are elements of usual energies: electrical and nuclear of atom P.

Let's introduce the following notations:

$A \equiv F(A)$ is se $^{A\equiv F(A)}$lf(A), $g(A) |||f(A)$ is se $^{g(A)\equiv f(A)}$lf(A).

self – type holistic leads to sself – type(designation).

parelf(sself – type).

singelf(sself – type).

The examples of self – type:

B ϵ B itself as element is self(B),

B ϵ B^2 itself as element is $sel_2f(B)$,

B ϵ f(B) itself as element is $sel_ff(B)$,

(…((B ϵ B)ϵB)ϵ…) itself as element,

B $(ϵ)^2$ B,

B f(ϵ) B,

B treats itself as any C,

A any action D itself,

(…((A any action D itself) any action D itself) any action D itself …) (*),

$$\begin{matrix}(*) & - & (*) \\ | & - & | \\ (*) & - & (*)\end{matrix},$$

$$\begin{matrix}ϵ & - & ϵ \\ | & - & | \\ ϵ & - & ϵ\end{matrix},$$

$$\begin{pmatrix}1, \begin{pmatrix}2, & 1\\ 3, & 2\end{pmatrix}\end{pmatrix},$$

$$\begin{pmatrix}1, \begin{matrix}2\\ 3\end{matrix}\end{pmatrix}$$

$$\begin{pmatrix}1, \begin{matrix}n_1\\ n_2\\ \dots\\ n_m\end{matrix}\end{pmatrix},$$

$$\begin{matrix}2 & - & 5 \\ | & - & | \\ 1 & - & 7\end{matrix},$$

$$\begin{matrix}|||^{-1} & - & ||| \\ | & - & |\end{matrix}.$$

$$\begin{matrix}os\mathit{elf} & - & s\mathit{elf}\end{matrix}$$

Remark 2.1.2.0. May consider any structure for interpretation form.

Projections of $\overset{A}{|||}\underset{C}{\overset{B}{}}$-interpretation by (2, 1)-interpretation:

1) A$|||_C$B-interpretation, 2) A$|||_B$C-interpretation, 3) C$|||_A$B-interpretation etc.

Projections of $\overset{A}{|||^{-1}}\underset{C}{\overset{B}{}}$-interpretation by (1, 2)-interpretation:

1) $|||_c^{-1}$(A, B)-interpretation, 2) $|||_A^{-1}$(C, B)-interpretation, 3) $|||_B^{-1}$(C, A)-interpretation etc.

(1, 2(2, 1)) = self^2 (see (1.1.4.1) [19]),

(1, (2, 1)) = self-point,

(2, 1) is (2, 1)-self,

(1, 2(1, 2)),

qulf = (1, 3, (1, 2)),

rulf = (1, (3, 1, 2)),

((1, 3), (1, 2))

(3, 1, 2) – interpretation,

(3, 1, 2)-self,

(3, 1, 2)-|||,

Self(|||) = |||(|||)|||,

(A, B, C) – interpretation,

SIIprt$\overset{A}{\underset{C}{B}}$- interpretation,

PrSIIprt$_B\overset{A}{\underset{C}{}}$- interpretation,

PrSIIprt$\overset{\{b_1,b_2\}}{\underset{R\quad C}{}}$- interpretation,

PrSIIprt$^{\{x_1, x_2, \ldots\}}_{X}$ is folding of space onto itself.

Remark 2.1.2.1. self(B) of the first type is energetic object with any B. In particular, on usual level self(living organism) of the first type gives manifestation by self of the second type has program DNA, which realizes its self so: by containing itself, it divides and thereby stops the internal cycle of containing its self, itself is no longer there, there are two other daughter DNAs, since this is already a lower level than the level of self.

Definition 2.1.2.1. Biself(A, B) is A contains into B as element and B contains into A as element simultaneously: Sprt $\begin{matrix} \text{Sprt}^{A}_{B} \\ \text{Sprt}^{B}_{A} \end{matrix}$ is interpretation (2, (4, 2))(A, B).

Definition 2.1.2.2. Nthself(A_1, A_2, …, A_N) is A_j contains A_i as element, ∀j, i = 1, 2, …, N simultaneously.

Definition 2.1.2.3. STRself(A) is any part Q of A contains any other part of A as element simultaneously.

Definition 2.1.2.4. STpRself(A) is any part Q of A contains any other part of A as element partially and simultaneously.

Definition 2.1.2.5. STp$_1$Rself(A) is any part Q of A partially contains any other part of A as element simultaneously.

Etc.

Definition 2.1.2.6. oBiself(A, B) is A expelling from B as element and B expelling from A as element and the first from the second simultaneously: $\begin{matrix} \text{Sprt}^{B}_{A} \\ \text{Sprt}^{B}_{A} \end{matrix}$ Sprt.

Definition 2.1.2.7. oNthself(A_1, A_2, …, A_N) is A_j contains and expelling from A_i as element, ∀j, i = 1, 2, …, N simultaneously.

Definition 2.1.2.8. oSTRself(A) is any part Q of A contains and expelling from any other part of A as element simultaneously.

Definition 2.1.2.9. oSTpRself(A) is any part Q of A contains and expelling from any other part of A as element partially and simultaneously.

Definition 2.1.2.10. oSTp$_1$Rself(A) is any part Q of A partially contains and partially expelling from any other part of A as element simultaneously.

Etc.

Definition 2.1.2.11. pBiself(A, B) is A expelling from B as element and B expelling from A as element and the first from the second simultaneously: $\begin{matrix}\mathrm{Sprt}_{A}^{B}\\ \mathrm{Sprt}_{A}^{B}\end{matrix}$ $\mathrm{Sprt}\begin{matrix}\mathrm{Sprt}_{B}^{A}\\ \mathrm{Sprt}_{A}^{B}\end{matrix}$.

Definition 2.1.2.12. pNthself(A_1, A_2, …, A_N) is A_j expelling from A_i as element, ∀j, i = 1, 2, …, N simultaneously.

Definition 2.1.2.13. pSTRself(A) is any part Q of A expelling from any other part of A as element simultaneously.

Definition 2.1.2.14. pSTpRself(A) is any part Q of A expelling from any other part of A as element partially and simultaneously.

Definition 2.1.2.15. pSTp$_1$Rself(A) is any part Q of A partially expelling from any other part of A as element simultaneously.

Etc.

2.1.3 Syntax of |||

A|||B is identification A into B, i.e., A and B become the same in the compression format (2, 1) or in the interpretation format (2, 1).

2.1.3.1 Measure of |||

$\mu_{|||}(A|||A^{-1}) = 1, \forall A,$

$\mu_{|||}(A|||B) = 1/2, \forall A, B: A \cap B = 0,$

$\mu_{|||}(A|||A) = 0, \forall A,$

$\mu_{|||}(A|||B) = ½ - \frac{(\mu(A \cap B) - \mu((-A) \cap B)}{2(\mu(A \cup B) + \mu((-A) \cup B)}, \forall A, B,$

$\mu(A|||B) = (\mu(A)\mu(B))^{(\mu_{|||}(A|||B) + 1)}$ (2.1*).

Some types of equations:

1. $\mu_{|||}(tw|||nmSprt) = \mu_{|||}(A|||x)$, tw|||nmSprt is activation of neurons set to fulfill target weights tw, x - ?.
2. $\mu_{|||}(tw|||nmSprt) = \mu_{|||}(y|||B)$, y - ?.
3. $\mu_{|||}(z|||nmSprt) = \mu_{|||}(A|||B)$, z - ?.
4. $\mu_{|||}((tw|||nmSprt)|||B^{tw}_{SprtA}) = \mu_{|||}(A|||x)$ with the analogue of Maxwell's equations for networks operating on electromagnetic energy:

   $rot(E^{tw}_{SprtA}) = -(1/ q_0)\, \partial(B^{tw}_{SprtA}) \ /\partial t, (*_H)$

   $rot(H^{tw}_{SprtA}) = j/q_0 + (1/ q_0) \cdot \partial\, (D^{tw}_{SprtA}) \ /\partial t, (**_H)$

   E^{tw}_{SprtA} - activation tension with target weight tw, B^{tw}_{SprtA} - induction of self with target weight tw, q_0- bioenergy constant, H^{tw}_{SprtA} – tension of self with target weight tw, D^{tw}_{SprtA} - 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 $x_1, x_2, \ldots, x_n$.
5. $\mu_{|||}((tw|||nmSprt)\ |||B^{tw}_{SprtA}) = \mu_{|||}(y|||Q)$

$\mathrm{rot}(E^{tw}_{SprtA}) = -(1/q_0)\,\partial(B^{tw}_{SprtA})\ /\partial t, (*_H)$

$\mathrm{rot}(H^{tw}_{SprtA}) = j/q_0 + (1/q_0)\cdot\partial\,(D^{tw}_{SprtA})\ /\partial t, (**_H)$, y - ?.

6. $\mu_{|||}((u|||nmSprt)\,|||B^{u}_{SprtA}) = \mu_{|||}(A|||Q)$

   $\mathrm{rot}(E^{u}_{SprtA}) = -(1/q_0)\,\partial(B^{u}_{SprtA})\ /\partial t, (*_H)$

   $\mathrm{rot}(H^{u}_{SprtA}) = j/q_0 + (1/q_0)\cdot\partial\,(D^{u}_{SprtA})\ /\partial t, (**_H)$, u - ?.

7. Etc.

Using (2.1*) we can rewrite these equations:

1. $\mu(tw|||nmSprt) = (\mu(tw)\mu(nmSprt))^{(\mu_{|||}(tw|||nmSprt)\,+\,1)} = (\mu(A)\mu(x))^{(\mu_{|||}(A|||x)+\,1)}$, x - ?.
2. $\mu(tw|||nmSprt) = (\mu(tw)\mu(nmSprt))^{(\mu_{|||}(tw|||nmSprt)\,+\,1)} = (\mu(y)\mu(B))^{(\mu_{|||}(y|||B)+\,1)}$, y - ?.
3. $\mu(z||nmSprt) = (\mu(z)\mu(nmSprt))^{(\mu_{|||}(z||nmSprt)\,+\,1)} = (\mu(A)\mu(B))^{(\mu_{|||}(A|||B)+\,1)}$, z - ?.
4. $(\mu(tw)\mu(nmSprt))^{(\mu_{|||}(tw|||nmSprt)\,+\,1)}\ \mu(B^{tw}_{SprtA}))^{(\mu_{|||}((tw|||nmSprt)\,||\,B^{tw}_{SprtA})+\,1)} = (\mu(A)\mu(x))^{(\mu_{|||}(A|||x)+\,1)}$ with the analogue of Maxwell's equations for networks operating on electromagnetic energy:

   $\mathrm{rot}(E^{tw}_{SprtA}) = -(1/q_0)\,\partial(B^{tw}_{SprtA})\ /\partial t, (*_H)$,

   $\mathrm{rot}(H^{tw}_{SprtA}) = j/q_0 + (1/q_0)\cdot\partial\,(D^{tw}_{SprtA})\ /\partial t, (**_H)$,

   E^{tw}_{SprtA} - activation tension with target weight tw, B^{tw}_{SprtA} - induction of self with target weight tw, q_0- bioenergy constant, H^{tw}_{SprtA} – tension of self with target weight tw, D^{tw}_{SprtA} - 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 $x_1, x_2, \ldots, x_n$.

5. $(\mu(\text{tw})\mu(\text{nmSprt}))^{(\mu_{|||}(\text{tw}|||\text{nmSprt})+1)}\ \mu(B^{tw}_{SprtA}))^{(\mu_{|||}((\text{tw}|||\text{nmSprt})\ ||\left|B^{tw}_{SprtA}\right.)+1)} = (\mu(y)\mu(Q))^{(\mu_{|||}(y|||Q)+1)}$,

   $\text{rot}(E^{tw}_{SprtA}) = -(1/q_0)\,\partial(B^{tw}_{SprtA})\ /\partial t, (*_H)$,

   $\text{rot}(H^{tw}_{SprtA}) = j/q_0 + (1/q_0)\cdot\partial\,(D^{tw}_{SprtA})\ /\partial t, (**_H)$, y - ?.

6. $(\mu(\text{u})\mu(\text{nmSprt}))^{(\mu_{|||}(\text{u}|||\text{nmSprt})+1)}\ \mu(B^{u}_{SprtA}))^{(\mu_{|||}((\text{tw}|||\text{nmSprt})\ ||\left|B^{u}_{SprtA}\right.)+1)} = (\mu(A)\mu(Q))^{(\mu_{|||}(A|||Q)+1)}$

   $\text{rot}(E^{u}_{SprtA}) = -(1/q_0)\,\partial(B^{u}_{SprtA})\ /\partial t, (*_H)$

   $\text{rot}(H^{u}_{SprtA}) = j/q_0 + (1/q_0)\cdot\partial\,(D^{u}_{SprtA})\ /\partial t, (**_H)$, u - ?.

7. Etc.

At activation of SmnSCprt it must be

$\mu(\ |||\text{mnSCprt}) \geq \mu(A\ |||B)$

for the replacement of A with B to take place at our usual level.

The relation of the possibility of implementing teleportation of human A into place x of space

$\mu(\text{self}(P(A)\ |||) \geq \mu(A\ |||x)$, P(A) - ?

The relation of the possibility of change R into V by self(P(A) |||)

$\mu(\text{self}(P(A)\ |||) \geq \mu(R\ |||V)$, P(A) - ?

The relation of the possibility of implementing telekinesis of E into place x of space

$\mu(\text{self}(P(A)\ |||) \geq \mu(E\ |||x)$, P(A) - ?

Etc.

Let's consider

$\overset{Z}{\rightarrow}$

$A|||B$,

Z, A, B – any,

|||(|||)||| = sel$f^{\frac{3}{2}}$(|||).

Kinds of |||

Hosts ||| **(**designation - |H||**).**

Penetrating ||| **(**designation - ||**P**|**).**

Examples:

| |P|$_{external}$ (|H||) | |P|$_{internal}$ for living organism,

||P| |H|| for others.

The assemblage point is a subject of ||| of the first kind, will is a subject of ||| of the second kind.

Some ||| **-***type*

A ||| A^{-1},

$-A$

$\sqrt{A}$

A ||| A^{-1},

$-A$

A|||$\sqrt{A}$,

A|||f(A),

2A|||A = self$\binom{2}{1}$(A), interpretation form is (1, ($\binom{2}{1}$, 1)),

A|||2A = self$\binom{1}{2}$(A), interpretation form is (1, ($\binom{1}{2}$, 1)),

F(A, 2A) ≡0,

$F(A)$

Q(A) ||| R(A).

$G(A)$

From point of view 2-interpretation;

$A|||_{x}B = self_{(A,\,B)}(x)$,

$A(pa|||_{x})B = paself_{(A,\,B)}(x)$.

$p_1paself(A) = A(pa|||)A^{-1}$,

$p_2paself(A) = A(pa|||)(-A)$,

$p_1aself(A) = A(|||)A^{-1}$,

$p_2aself(A) = A(|||)(-A)$,

$ppself_{(A,B)}(x) = AQB$, $Q = Sprt_{x}^{\left\{Sprt_{x}^{\{\ ,\ \}},Sprt_{x}^{\{\ ,\ \}}\right\}}$,

$pa|||_{A}$ -double of A.

A||| - double of A.

self(A|||) - double of A.

$A|||_{x} = self_{(A,\ )}(x)$ - double of A at x.

$Apa|||_{x} = paself_{(A,\ )}(x)$ - double of A at x.

$papa||| = pa|||(\ pa|||)\ (pa|||)^{-1}$.

$|||^{-1}(A|||B) = A, B$.

$(\sum_{i=1}^{N} A_i)||| (\sum_{i=1}^{N} A_i^{-1})$,

$(\sum_{i=1}^{N} A_i)||| ((\sum_{i=1}^{N} A_i)^{-\mathbf{1}}$,

$self((\sum_{i=1}^{N} self - type\ A_i)(\sum_{i=1}^{N} ||| - type\ B_i)$,

self − type(|| |− type), where (||| − type) – argument.

$|||_{Q}^{-\mathbf{1}}C$, for example, $|||_{(A,\,B)}^{-\mathbf{1}}C = C_A, C_B$,

${}_{a}^{a}Sprt_{a}^{a}$ = oself(a)|||self(a) and paself by containment,

$oself(a) \neq (self(a))^{-1}$,

$(self(a))^{-1} = a$,

a↑ | ↓a – self by a and paself by action,

$S|||_{g}^{A}B$ is A||| with B by target weight g,

$A|||S\overset{f}{\underset{u}{}}B = ((|||\uparrow_u)\ |||\downarrow_f)$,

$d_x(|||)(-d_y)$, x, y are different space places.

Let us denote a singularity of type $D|||_\infty D$ by s∞elf(D), e.g., s∞elf(set) = the set of all sets.

pas∞elf(D) = $(D|||_\infty^{-1} D)\ |||\ (D|||_\infty D)$,

$Sprt\overset{\{A,\ B\}}{\underset{c}{}} = A|||_c B$,

$\begin{matrix} q_n \\ ||| \cdots \\ q_1 \end{matrix}$,

$\begin{matrix} q_B \\ ||| \cdots \\ q_A \end{matrix}$,

$\begin{matrix} \{any\ C\} \\ ||| \quad \cdots \quad \\ \{any\ D\} \end{matrix}$,

$\begin{matrix} q_n \\ sel \cdots f, \\ q_1 \end{matrix}$

parelf(sself − type),

parelf(|||),

$parelf\overset{Z}{\rightarrow}$ parelf,

$parelf\overset{Z}{\rightarrow}$singelf,

$singelf\overset{Z}{\rightarrow}$parelf,

$singelf\overset{Z}{\rightarrow}$singelf,

$parel_N(|||)$,

$parel_N(parel_N(|||))$,

$pasel_N(pa|||)$,

$sel_N(|||)$.

Remark 2.1.3.1. May consider self-hierarchy of interpretations, self-type hierarchy of interpretations, |||-type hierarchy of interpretations, Spiral |||, Spiral pa|||.

Remark 2.1.3.2. A measure can be introduced for any dynamic operator.

Remark 2.1.3.3. For example, for an organ:

interpretation (((1, (2, 1)), (2, (1, (2, (1, (2, 1))))), for an organism interpretation

$$\begin{pmatrix} (1,\ (2,1)),\ (2,\ \ (1,\ (2,\ (1,\ (2,\ (1,\ (2,1)))))))) \\ (1,\ (2,1)),\ (1,\ (1,\ 2(1,\ 2(1,\ 2(1,\ (1,\ 2(1,2)))))))) \end{pmatrix}.$$

Remark 2.1.3.4. The actions of a short-pulse laser has the following structures: ||| self(|||), ↑ | ↓|||self(|||), ↑ | ↓|||.

Ordered paself

$\overrightarrow{paself(d)} = d_x(\vec{|||})(-d_y)$ (),

x, y are in particular, different space places or x =y.

Here d is in different space places: by the form d in x - d_x and the form (-d) - ($-d_y$) simultaneously. d may be a program. Then $\overrightarrow{paself(d)}$ will be ordered paself program.

Ordered |||

A$\vec{|||}$ B is A ||| into B, but not vice versa (but not on the contrary).

Remark 2.1.3.5. Codes for self are 1, ||| - 0, and binary arithmetic can be used for programming. For example,self + ||| + $self^3$ + $|||^6$. For normal manipulation: self + $self^3$ +

||| + $|||^6$. For self- manipulation:

Nano-(space-time) by $|||^{-1}$, ${}^{q}_{q}$Sprt, Nano-elements by $|||^{-1}$, ${}^{q}_{q}$Sprt, Nano-any C by $|||^{-1}$, ${}^{q}_{q}$Sprt. Briefly pulsed laser as a light identifyer can nano-disidentify matter.

Ultraviolet UVC radiation is closer to self, UHF is closer to paself

Any object has a self-type, |||-type root, of which it is a manifestation. The CNS has its own self-type, |||-type root.

Remark 2.1.3.6. The usual 2-interpretation comes from the nature of the mind, which is determined by the nature of the internal dialogue, i.e., the activation of B (in particular, the cause) activates C (the effect). In the case of other interpretations (e.g., N-interpretations, N>2), e.g., through the will, all sorts of activations of various elements of the interpretation structure are possible, which is what this interpretation will consist of.

Remark 2.1.3.7. For N-interpretations, N>2, a generalization of the form (1.1) for self, for example,

$$\begin{pmatrix} 3 & & 5 & 7 & 2 \\ 2 & 1 & 3 & 0 & 1 \end{pmatrix}.$$

Remark 2.1.3.8. Dynamic operators can be used as interpretations forms.

Remark 2.1.3.9. $\begin{pmatrix} R||| \\ Q||| \end{pmatrix}\mathrm{D}\begin{pmatrix} (g_2)_{C|\mathbf{g_1}} \\ g_3 \end{pmatrix}$ - R is shaded by D.

2. 2 Elements of mathematical theory of fuzzy interpretations

We will consider the following interpretations forms

$\tilde{y} = (y_1|\mu_{\tilde{y}}(y_1), y_2|\mu_{\tilde{y}}(y_2))$,

where $\tilde{y}$ is the fuzzy set with measure of fuzziness $\mu_{\tilde{y}}$, y_1, y_2 are any fuzzy elements of its.

2. 2.1 Simplest operations with fuzzy interpretations forms by containment

Addition operation with fuzzy interpretation forms:

$(y_1|\mu_{\tilde{y}}(y_1), y_2|\mu_{\tilde{y}}(y_2)) +_{in} (b_1|\mu_{\tilde{b}}(b_1), b_2|\mu_{\tilde{b}}(b_2)) = (y_1|\mu_{\tilde{y}}(y_1) + b_1|\mu_{\tilde{b}}(b_1), y_2|\mu_{\tilde{y}}(y_2) + b_2|\mu_{\tilde{b}}(b_2))$,

$y_1|\mu_{\tilde{y}}(y_1)$, $b_1|\mu_{\tilde{b}}(b_1)$ are homogeneous any, $y_2|\mu_{\tilde{y}}(y_2)$, $b_2|\mu_{\tilde{b}}(b_2)$ are homogeneous any.

Remark 2. 2.1.1. In particular, $y_1|\mu_{\tilde{y}}(y_1)$ and $b_1|\mu_{\tilde{b}}(b_1)$ can be other structurally homogeneous forms. The same applies to $y_2|\mu_{\tilde{y}}(y_2)$ and $b_2|\mu_{\tilde{b}}(b_2)$.

The following relationships are held:

1) Commutative law of addition
   $(y_1|\mu_{\tilde{y}}(y_1), y_2|\mu_{\tilde{y}}(y_2)) +_{in} (b_1|\mu_{\tilde{b}}(b_1), b_2|\mu_{\tilde{b}}(b_2)) = (b_1|\mu_{\tilde{b}}(b_1), b_2|\mu_{\tilde{b}}(b_2)) +_{in} (y_1|\mu_{\tilde{y}}(y_1), y_2|\mu_{\tilde{y}}(y_2))$.
2) Associative law of addition
   $((y_1|\mu_{\tilde{y}}(y_1), y_2|\mu_{\tilde{y}}(y_2)) +_{in} (b_1|\mu_{\tilde{b}}(b_1), b_2|\mu_{\tilde{b}}(b_2))) +_{in} (b_3|\mu_{\tilde{b}}(b_3), b_4|\mu_{\tilde{b}}(b_4)) = (y_1|\mu_{\tilde{y}}(y_1), y_2|\mu_{\tilde{y}}(y_2)) +_{in} ((b_1|\mu_{\tilde{b}}(b_1), b_2|\mu_{\tilde{b}}(b_2)) +_{in} (b_3|\mu_{\tilde{b}}(b_3), b_4|\mu_{\tilde{b}}(b_4)))$.

The subtraction operation with fuzzy interpretation forms:

$(y_1|\mu_{\tilde{y}}(y_1), y_2|\mu_{\tilde{y}}(y_2)) -_{in} (b_1|\mu_{\tilde{b}}(y_1), b_2|\mu_{\tilde{b}}(y_2)) = (y_1|\mu_{\tilde{y}}(y_1) - b_1|\mu_{\tilde{b}}(y_1), y_2|\mu_{\tilde{y}}(y_2) - b_2|\mu_{\tilde{b}}(y_2))$.

The multiplication operation with fuzzy interpretation forms:

$(y_1|\mu_{\tilde{y}}(y_1), y_2|\mu_{\tilde{y}}(y_2)) *_{in} (b_1|\mu_{\tilde{b}}(b_1), b_2|\mu_{\tilde{b}}(b_2)) = (y_1|\mu_{\tilde{y}}(y_1) * b_1|\mu_{\tilde{b}}(b_1), y_2|\mu_{\tilde{y}}(y_2) * b_2|\mu_{\tilde{b}}(b_2))$, $y_1|\mu_{\tilde{y}}(y_1)$, $b_1|\mu_{\tilde{b}}(b_1)$ are homogeneous any, $y_2|\mu_{\tilde{y}}(y_2)$, $b_2|\mu_{\tilde{b}}(b_2)$ are homogeneous any.

Remark 2. 2.1.2. In particular, $y_1|\mu_{\tilde{y}}(y_1)$ and $b_1|\mu_{\tilde{b}}(b_1)$ can be other structurally homogeneous forms. The same applies to $y_2|\mu_{\tilde{y}}(y_2)$ and $b_2|\mu_{\tilde{b}}(b_2)$.

The following relationships hold:

1) Commutative law of multiplication

2) $(y_1|\mu_{\tilde{y}}(y_1), y_2|\mu_{\tilde{y}}(y_2)) *_{in} (b_1|\mu_{\tilde{b}}(b_1), b_2|\mu_{\tilde{b}}(b_2)) = (b_1|\mu_{\tilde{b}}(b_1), b_2|\mu_{\tilde{b}}(b_2)) *_{in} (y_1|\mu_{\tilde{y}}(y_1), y_2|\mu_{\tilde{y}}(y_2))$.

3) Associative law of multiplication

4) $((y_1|\mu_{\tilde{y}}(y_1), y_2|\mu_{\tilde{y}}(y_2)) *_{in} (b_1|\mu_{\tilde{b}}(b_1), b_2|\mu_{\tilde{b}}(b_2))) *_{in} (b_3|\mu_{\tilde{b}}(b_3), b_4|\mu_{\tilde{b}}(b_4)) = (y_1|\mu_{\tilde{y}}(y_1), y_2|\mu_{\tilde{y}}(y_2)) *_{in} ((b_1|\mu_{\tilde{b}}(b_1), b_2|\mu_{\tilde{b}}(b_2)) *_{in} (b_3|\mu_{\tilde{b}}(b_3), b_4|\mu_{\tilde{b}}(b_4)))$.

5) Distributive law of multiplication with respect to addition

$((y_1|\mu_{\tilde{y}}(y_1), y_2|\mu_{\tilde{y}}(y_2)) +_{in} (b_1|\mu_{\tilde{b}}(b_1), b_2|\mu_{\tilde{b}}(b_2))) *_{in} (b_3|\mu_{\tilde{b}}(b_3), b_4|\mu_{\tilde{b}}(b_4)) = (y_1|\mu_{\tilde{y}}(y_1), y_2|\mu_{\tilde{y}}(y_2)) *_{in} (b_3|\mu_{\tilde{b}}(b_3), b_4|\mu_{\tilde{b}}(b_4)) +_{in} (b_1|\mu_{\tilde{b}}(b_1), b_2|\mu_{\tilde{b}}(b_2)) *_{in} (b_3|\mu_{\tilde{b}}(b_3), b_4|\mu_{\tilde{b}}(b_4))$

6) Distributive law of multiplication with respect to subtraction

7) $((y_1|\mu_{\tilde{y}}(y_1), y_2|\mu_{\tilde{y}}(y_2)) -_{in} (b_1|\mu_{\tilde{b}}(b_1), b_2|\mu_{\tilde{b}}(b_2))) *_{in} (b_3|\mu_{\tilde{b}}(b_3), b_4|\mu_{\tilde{b}}(b_4)) = (y_1|\mu_{\tilde{y}}(y_1), y_2|\mu_{\tilde{y}}(y_2)) *_{in} (b_3|\mu_{\tilde{b}}(b_3), b_4|\mu_{\tilde{b}}(b_4)) -_{in} (b_1|\mu_{\tilde{b}}(b_1), b_2|\mu_{\tilde{b}}(b_2)) *_{in} (b_3|\mu_{\tilde{b}}(b_3), b_4|\mu_{\tilde{b}}(b_4))$

8) $(y_1|\mu_{\tilde{y}}(y_1), y_2|\mu_{\tilde{y}}(y_2)) *_{in} (1, 1) = (y_1|\mu_{\tilde{y}}(y_1), y_2|\mu_{\tilde{y}}(y_2))$

From $(y_1|\mu_{\tilde{y}}(y_1), y_2|\mu_{\tilde{y}}(y_2)) +_{in} (b_1|\mu_{\tilde{b}}(b_1), b_2|\mu_{\tilde{b}}(b_2)) = (b_3|\mu_{\tilde{b}}(b_3), b_4|\mu_{\tilde{b}}(b_4)) +_{in} (b_1|\mu_{\tilde{b}}(b_1), b_2|\mu_{\tilde{b}}(b_2))$ follows $(y_1|\mu_{\tilde{y}}(y_1), y_2|\mu_{\tilde{y}}(y_2)) = (b_3|\mu_{\tilde{b}}(b_3), b_4|\mu_{\tilde{b}}(b_4))$

9) From $(y_1|\mu_{\tilde{y}}(y_1), y_2|\mu_{\tilde{y}}(y_2)) *_{in} (b_1|\mu_{\tilde{b}}(b_1), b_2|\mu_{\tilde{b}}(b_2)) = (b_3|\mu_{\tilde{b}}(b_3), b_4|\mu_{\tilde{b}}(b_4)) *_{in} (b_1|\mu_{\tilde{b}}(b_1), b_2|\mu_{\tilde{b}}(b_2))$ follows $(y_1|\mu_{\tilde{y}}(y_1), y_2|\mu_{\tilde{y}}(y_2)) = (b_3|\mu_{\tilde{b}}(b_3), b_4|\mu_{\tilde{b}}(b_4))$.

The division operation with fuzzy interpretation forms:

$(y_1|\mu_{\tilde{y}}(y_1), y_2|\mu_{\tilde{y}}(y_2)) /_{in} (b_1|\mu_{\tilde{b}}(b_1), b_2|\mu_{\tilde{b}}(b_2))$ or $\frac{(y_1|\mu_{\tilde{y}}(y_1), y_2|\mu_{\tilde{y}}(y_2))}{(b_1|\mu_{\tilde{b}}(b_1), b_2|\mu_{\tilde{b}}(b_2))}_{in}$, $b_1|\mu_{\tilde{b}}(b_1) \neq 0$, $b_2|\mu_{\tilde{b}}(b_2), \neq 0$.

$(y_1|\mu_{\tilde{y}}(y_1), y_2|\mu_{\tilde{y}}(y_2)) /_{in} (b_1|\mu_{\tilde{b}}(b_1), b_2|\mu_{\tilde{b}}(b_2)) = (y_1|\mu_{\tilde{y}}(y_1) / b_1|\mu_{\tilde{b}}(b_1), y_2|\mu_{\tilde{y}}(y_2) / b_2|\mu_{\tilde{b}}(b_2)) = (\frac{y_1|\mu_{\tilde{y}}(y_1)}{(b_1|\mu_{\tilde{b}}(b_1)}_{in}, \frac{y_2|\mu_{\tilde{y}}(y_2)}{(b_2|\mu_{\tilde{b}}(b_2)}_{in})$, $b_1|\mu_{\tilde{b}}(b_1) \neq 0$, $b_2|\mu_{\tilde{b}}(b_2), \neq 0$.

The fundamental property of a fraction

$$\frac{(y_1|\mu_{\tilde{y}}(y_1),\, y_2|\mu_{\tilde{y}}(y_2))*_{in}(b_3|\mu_{\tilde{b}}(b_3),\, b_4|\mu_{\tilde{b}}(b_4))}{(b_1|\mu_{\tilde{b}}(b_1),\, b_2|\mu_{\tilde{b}}(b_2))*_{in}(b_3|\mu_{\tilde{b}}(b_3),\, b_4|\mu_{\tilde{b}}(b_4))}_{\text{in}} = \frac{(y_1|\mu_{\tilde{y}}(y_1),\, y_2|\mu_{\tilde{y}}(y_2))}{(b_1|\mu_{\tilde{b}}(b_1),\, b_2|\mu_{\tilde{b}}(b_2))}_{\text{in}},\ b_1|\mu_{\tilde{b}}(b_1) \neq 0,\ b_2|\mu_{\tilde{b}}(b_2) \neq 0,\ b_3|\mu_{\tilde{b}}(b_1) \neq 0,\ b_4|\mu_{\tilde{b}}(b_2) \neq 0.$$

Operations with fractions

$$\frac{(y_1|\mu_{\tilde{y}}(y_1),\, y_2|\mu_{\tilde{y}}(y_2))}{(b_1|\mu_{\tilde{b}}(b_1),\, b_2|\mu_{\tilde{b}}(b_2))}_{\text{in}} +_{\text{in}} \frac{(b_3|\mu_{\tilde{b}}(b_3),\, b_4|\mu_{\tilde{b}}(b_4))}{(b_1|\mu_{\tilde{b}}(b_1),\, b_2|\mu_{\tilde{b}}(b_2))}_{\text{in}} = \frac{(y_1|\mu_{\tilde{y}}(y_1),\, y_2|\mu_{\tilde{y}}(y_2))+_{in}(b_3|\mu_{\tilde{b}}(b_3),\, b_4|\mu_{\tilde{b}}(b_4))}{(b_1|\mu_{\tilde{b}}(b_1),\, b_2|\mu_{\tilde{b}}(b_2))}$$

$$_{\text{in}},\ b_1|\mu_{\tilde{b}}(b_1) \neq 0,\ b_2|\mu_{\tilde{b}}(b_2) \neq 0,$$

$$\frac{(y_1|\mu_{\tilde{y}}(y_1),\, y_2|\mu_{\tilde{y}}(y_2))}{(b_1|\mu_{\tilde{b}}(b_1),\, b_2|\mu_{\tilde{b}}(b_2))}_{\text{in}} +_{\text{in}} \frac{(b_3|\mu_{\tilde{b}}(b_3),\, b_4|\mu_{\tilde{b}}(b_4))}{(b_5|\mu_{\tilde{b}}(b_5),\, b_6|\mu_{\tilde{b}}(b_6))}_{\text{in}} =$$

$$\frac{(y_1|\mu_{\tilde{y}}(y_1),\, y_2|\mu_{\tilde{y}}(y_2))*_{in}(S,\, P)+_{in}(b_3|\mu_{\tilde{b}}(b_3),\, b_4|\mu_{\tilde{b}}(b_4))*_{in}(b_1|\mu_{\tilde{b}}(b_1),\, b_2|\mu_{\tilde{b}}(b_2))}{(b_1|\mu_{\tilde{b}}(b_1),\, b_2|\mu_{\tilde{b}}(b_2))*_{in}(b_5|\mu_{\tilde{b}}(b_5),\, b_6|\mu_{\tilde{b}}(b_6))}_{\text{in}},\ b_1|\mu_{\tilde{b}}(b_1) \neq 0,\ b_2|\mu_{\tilde{b}}(b_2) \neq 0,\ b_5|\mu_{\tilde{b}}(b_5) \neq 0,\ b_6|\mu_{\tilde{b}}(b_6) \neq 0.$$

$$\frac{(y_1|\mu_{\tilde{y}}(y_1),\, y_2|\mu_{\tilde{y}}(y_2))}{(b_1|\mu_{\tilde{b}}(b_1),\, b_2|\mu_{\tilde{b}}(b_2))}_{\text{in}} *_{\text{in}} \frac{(b_3|\mu_{\tilde{b}}(b_3),\, b_4|\mu_{\tilde{b}}(b_4))}{(b_5|\mu_{\tilde{b}}(b_5),\, b_6|\mu_{\tilde{b}}(b_6))}_{\text{in}} = \frac{(y_1|\mu_{\tilde{y}}(y_1),\, y_2|\mu_{\tilde{y}}(y_2))*_{in}(b_3|\mu_{\tilde{b}}(b_3),\, b_4|\mu_{\tilde{b}}(b_4))}{(b_1|\mu_{\tilde{b}}(b_1),\, b_2|\mu_{\tilde{b}}(b_2))*_{in}(b_5|\mu_{\tilde{b}}(b_5),\, b_6|\mu_{\tilde{b}}(b_6))}_{\text{in}},$$

$$\frac{(y_1|\mu_{\tilde{y}}(y_1),\, y_2|\mu_{\tilde{y}}(y_2))}{(b_1|\mu_{\tilde{b}}(b_1),\, b_2|\mu_{\tilde{b}}(b_2))}_{\text{in}} /_{\text{in}} \frac{(b_3|\mu_{\tilde{b}}(b_3),\, b_4|\mu_{\tilde{b}}(b_4))}{(b_5|\mu_{\tilde{b}}(b_5),\, b_6|\mu_{\tilde{b}}(b_6))}_{\text{in}} = \frac{(y_1|\mu_{\tilde{y}}(y_1),\, y_2|\mu_{\tilde{y}}(y_2))*_{in}(b_5|\mu_{\tilde{b}}(b_5),\, b_6|\mu_{\tilde{b}}(b_6))}{(b_1|\mu_{\tilde{b}}(b_1),\, b_2|\mu_{\tilde{b}}(b_2))*_{in}(b_3|\mu_{\tilde{b}}(b_3),\, b_4|\mu_{\tilde{b}}(b_4))}_{\text{in}},$$

$$\left((y_1|\mu_{\tilde{y}}(y_1),\, y_2|\mu_{\tilde{y}}(y_2)) /_{in} (b_1|\mu_{\tilde{b}}(b_1),\, b_2|\mu_{\tilde{b}}(b_2))\right)^n = \frac{(y_1|\mu_{\tilde{y}}(y_1),\, y_2|\mu_{\tilde{y}}(y_2))^n}{(b_1|\mu_{\tilde{b}}(b_1),\, b_2|\mu_{\tilde{b}}(b_2))^n}_{\text{in}},$$

$$b_1|\mu_{\tilde{b}}(b_1) \neq 0,\ b_2|\mu_{\tilde{b}}(b_2) \neq 0.$$

The following relationship holds:

$$\sqrt{(y_1|\mu_{\tilde{y}}(y_1),\, y_2|\mu_{\tilde{y}}(y_2))} = (\sqrt{(y_1|\mu_{\tilde{y}}(y_1)}\,,\, \sqrt{(y_2|\mu_{\tilde{y}}(y_2)}\,).$$

2. 2. 2 Syntax of fuzzy self-like formations

The usual fuzzy self(D) = D(($\tilde{|||}$)|$\mu_{\tilde{|||}}$)D(designation fself(D)) and corresponds to the next higher level of the hierarchy and the fuzzy form (1, (2, 1)), ($\tilde{|||}$)|$\mu_{\tilde{|||}}$ is fuzzy ||| with measure of fuzziness $\mu_{\tilde{|||}}$.

General definition of self-type:

A $\overset{Z}{\rightarrow}$ itself, Z is any fuzzy.

If in the relation defining the fuzzy automorphism we replace the "=" sign with the "≡" sign, we obtain the corresponding form of fself.

Let's introduce the following concept of dfself(A), A = $A_1 \cup A_2$, which consist from fuzzy elements of some hierarchical levels: by self-level corresponds to fuzzy element *a* from A_1, which is capable of fuzzy generating A, and by |||-level corresponds to A_2 = Q*self(P(($\tilde{|||}$)|$\mu_{\tilde{|||}}$)), which has the embryo itself in the form of a compression of itself: R*paself(S(pa($\tilde{|||}$)|$\mu_{\tilde{|||}}$)) consists from all positions of the assemblage point and the compression of the assemblage point itself into one point and is equal $\overset{w}{\underset{Q*\,fself(P((\tilde{|||})|\mu_{\tilde{|||}}))}{\overset{\mu_2}{}}}$ ffSprt $\overset{Q*\,fself(P((\tilde{|||})|\mu_{\tilde{|||}}))}{\underset{w}{\overset{\mu_1}{}}}$, where Q* fself(P(($\tilde{|||}$)|$\mu_{\tilde{|||}}$)) fuzzy fits into w with measure of fuzziness μ_1, Q* fself(P(($\tilde{|||}$)|$\mu_{\tilde{|||}}$)) is fuzzy forced out of w with measure of fuzziness μ_2 simultaneously[1]. Then for FVprt – element of human A [20]

$$\left(\begin{array}{c} \overset{\cdots}{} \\ parel f A\left(decignation - \overline{\overline{\overline{A}}}\right) \\ singel f A\left(decignation - \overline{\overline{\overline{A}}}\right) \\ subtle\ energy\ of\ object\ \ A\ \text{paradoxical}\ upper\ level\ (\ pa|||)\left(decignation - \overline{\overline{\overline{A}}}\right) \\ subtle\ energy\ of\ object\ \ A\ \text{paradoxical}\ mid-level\left(decignation - \overline{\overline{\overline{A}}}\right) \\ \begin{array}{cc} / & \backslash \\ subtle\ energy\ of\ |||^{-1} & subtle\ energy\ of\ ||| \\ \underline{\underline{\underline{A}}} & \overline{\overline{\overline{A}}} \\ \left(\underline{\underline{A}}\right) & \left(\overline{\overline{A}}\right) \end{array} \\ ordinary\ energy\ exhibited\ by\ an\ \ object\ \ A\left(decignation - \underline{\underline{A}}\right) \leftarrow \text{the raw energy of an object}\ \ A\left(decignation - A\right) \end{array}\right) (**)$$

$\overline{\overline{\overline{A}}} = a$, $\overline{\overline{\overline{A}}}$ = Q* fself(P(($\tilde{|||}$)|$\mu_{\tilde{|||}}$)).

Let's introduce the following notations:

A≡ F(A) is fse $^{A\equiv F(A)}$lf(A), g(A) |||f(A) is fse $^{g(A)\equiv f(A)}$lf(A).

fself − type holistic leads to fsself − type(designation)

parelf(fsself − type)

singelf(fsself − type)

The examples of fself − type:

fuzzy B ϵ fuzzy B itself as element is fself(B),

fuzzy B ϵ fuzzy B^2 itself as element is $fsel_2f(B)$,

fuzzy B ϵ fuzzy f(B) itself as element is $fsel_ff(B)$,

(…((fuzzy B ϵ fuzzy B) ϵ fuzzy B)ϵ…) itself as element,

fuzzy B (fuzzy ϵ)^2 fuzzy B,

fuzzy B f(fuzzy ϵ) fuzzy B,

fuzzy B fuzzy treats itself as any C,

fuzzy A any fuzzy action D itself,

(…((fuzzy A any fuzzy action D itself) any fuzzy action D itself) any fuzzy action D itself …) (*),

$$\begin{matrix}(*) & - & (*)\\ | & - & | \\ (*) & - & (*)\end{matrix},$$

Any fuzzy structure for fuzzy interpretation form,

$fSelf(((|\tilde{\tilde{|}}|)|\mu_{\tilde{\tilde{|}}|}) = ((|\tilde{\tilde{|}}|)|\mu_{\tilde{\tilde{|}}|})(((|\tilde{\tilde{|}}|)|\mu_{\tilde{\tilde{|}}|}))((|\tilde{\tilde{|}}|)|\mu_{\tilde{\tilde{|}}|})$,

(fuzzy A, fuzzy B, fuzzy C) – interpretation,

$fSIIprt\begin{matrix}A\\B\\C\end{matrix}$- interpretation [],

$fPrSIIprt_{B}^{\ A}{}_{C}$- interpretation,

$fPrSIIprt_{R\ \ C}^{\{b_1,b_2\}}$- interpretation,

$fPrSIIprt_{X}^{\{x_1,x_2,\ \ldots\}}$ is fuzzy folding of space onto itself.

2. 2.3 Syntax of fuzzy |||

$A((|\tilde{|}|)|\mu_{\tilde{|}|})\overset{Z}{\rightarrow}$B,

Z, A, B – any,

$((|\tilde{|}|)|\mu_{\tilde{|}|})(((|\tilde{|}|)|\mu_{\tilde{|}|}))((|\tilde{|}|)|\mu_{\tilde{|}|}) = \mathbf{f}\mathrm{sel}f^{\frac{3}{2}}((|\tilde{|}|)|\mu_{\tilde{|}|})$.

Kinds of $(|\tilde{|}|)|\mu_{\tilde{|}|}$

Hosts $(|\tilde{|}|)|\mu_{\tilde{|}|}$ (designation - **|fH||),**

Penetrating $(|\tilde{|}|)|\mu_{\tilde{|}|}$ (designation - **||fP|).**

Examples:

$|\,|\mathrm{fP}|_{external}$ (|fH||) $|\,|\mathrm{fP}|_{internal}$ for living organism,

||fP| |fH|| for others.

Some $(|\tilde{|}|)|\mu_{\tilde{|}|}$ *-type*

$\mathrm{A}(|\tilde{|}|)|\mu_{\tilde{|}|}\mathrm{A}^{-1}$,

$-A$

$\sqrt{A}$

$\mathrm{A}(|\tilde{|}|)|\mu_{\tilde{|}|}\mathrm{A}^{-1}$,

$-A$

$\mathrm{A}(|\tilde{|}|)|\mu_{\tilde{|}|}\sqrt{A}$,

$\mathrm{A}(|\tilde{|}|)|\mu_{\tilde{|}|}\mathrm{f}(\mathrm{A})$,

$F(A)$

$\mathrm{Q}(\mathrm{A})(|\tilde{|}|)|\mu_{\tilde{|}|}\mathrm{R}(\mathrm{A})$,

$G(A)$

$\mathrm{p}_1\mathrm{pafself}(\mathrm{A}) = \mathrm{A}(\mathrm{pa}(|\tilde{|}|)|\mu_{\tilde{|}|})\mathrm{A}^{-1}$,

$\mathrm{p}_2\mathrm{pafself}(\mathrm{A}) = \mathrm{A}(\mathrm{pa}(|\tilde{|}|)|\mu_{\tilde{|}|})(-\mathrm{A})$,

$p_1$afself(A) = A((|$\tilde{\tilde{\tilde{|}}}$|)|$\mu_{\tilde{\tilde{\tilde{|}}}}$)$A^{-1}$,

$p_2$afself(A) = A((|$\tilde{\tilde{\tilde{|}}}$|)|$\mu_{\tilde{\tilde{\tilde{|}}}}$)(-A),

papaf(|$\tilde{\tilde{\tilde{|}}}$|)|$\mu_{\tilde{\tilde{\tilde{|}}}}$ = pa(|$\tilde{\tilde{\tilde{|}}}$|)|$\mu_{\tilde{\tilde{\tilde{|}}}}$(pa(|$\tilde{\tilde{\tilde{|}}}$|)|$\mu_{\tilde{\tilde{\tilde{|}}}}$) (pa(|$\tilde{\tilde{\tilde{|}}}$|)|$\mu_{\tilde{\tilde{\tilde{|}}}}$)$^{-1}$.

(|$\tilde{\tilde{\tilde{|}}}$|)|$\mu_{\tilde{\tilde{\tilde{|}}}}$)$^{-1}$(A(|$\tilde{\tilde{\tilde{|}}}$|)|$\mu_{\tilde{\tilde{\tilde{|}}}}$B) = A, B.

$(\sum_{i=1}^{N} A_i)$(|$\tilde{\tilde{\tilde{|}}}$|)|$\mu_{\tilde{\tilde{\tilde{|}}}}$ $(\sum_{i=1}^{N} A_i^{-1})$,

$(\sum_{i=1}^{N} A_i)$(|$\tilde{\tilde{\tilde{|}}}$|)|$\mu_{\tilde{\tilde{\tilde{|}}}}$ $((\sum_{i=1}^{N} A_i)^{-1}$,

fself$((\sum_{i=1}^{N}$ fself – type $A_i)(\sum_{i=1}^{N}$ (|$\tilde{\tilde{\tilde{|}}}$|)|$\mu_{\tilde{\tilde{\tilde{|}}}}$ – type $B_i)$,

fself – type((|$\tilde{\tilde{\tilde{|}}}$|)|$\mu_{\tilde{\tilde{\tilde{|}}}}$ – type, ((|$\tilde{\tilde{\tilde{|}}}$|)|$\mu_{\tilde{\tilde{\tilde{|}}}}$ – type) – argument.

(|$\tilde{\tilde{\tilde{|}}}$|)|$\mu_{\tilde{\tilde{\tilde{|}}}}$)$_Q^{-1}$C, for example, (|$\tilde{\tilde{\tilde{|}}}$|)|$\mu_{\tilde{\tilde{\tilde{|}}}}$)$_{(A,\,B)}^{-1}$= C_A, C_B,

${}_a^a fSprt_a^a$ = foself(a)|||fself(a) and pafself by containment,

foself(a) ≠ (fself(a))$^{-1}$,

$S(|\tilde{\tilde{\tilde{|}}}|)|\mu_{\tilde{\tilde{\tilde{|}}}}{}^{A}_{B}_{g}$ is A(|$\tilde{\tilde{\tilde{|}}}$|)|$\mu_{\tilde{\tilde{\tilde{|}}}}$ with B by target weight g,

A(|$\tilde{\tilde{\tilde{|}}}$|)|$\mu_{\tilde{\tilde{\tilde{|}}}}S_u^f$B = (((|$\tilde{\tilde{\tilde{|}}}$|)|$\mu_{\tilde{\tilde{\tilde{|}}}}\uparrow_u$) (|$\tilde{\tilde{\tilde{|}}}$|)|$\mu_{\tilde{\tilde{\tilde{|}}}}\downarrow_f$),

d_x((|$\tilde{\tilde{\tilde{|}}}$|)|$\mu_{\tilde{\tilde{\tilde{|}}}}$)(-$d_y$), x, y are different space places.

Let us denote a singularity of type D((|$\tilde{\tilde{\tilde{|}}}$|)|$\mu_{\tilde{\tilde{\tilde{|}}}}$)$_\infty$D by fs∞elf(D), e.g., fs∞elf(set) = the fuzzy set of all fuzzy sets.

pafs∞elf(D) = (D((|$\tilde{\tilde{\tilde{|}}}$|)|$\mu_{\tilde{\tilde{\tilde{|}}}}$)$_\infty^{-1}$D) (|$\tilde{\tilde{\tilde{|}}}$|)|$\mu_{\tilde{\tilde{\tilde{|}}}}$ (D((|$\tilde{\tilde{\tilde{|}}}$|)|$\mu_{\tilde{\tilde{\tilde{|}}}}$)$_\infty$D),

$fSprt^{\{A,\,B\}}_{C}$ = A((|$\tilde{\tilde{\tilde{|}}}$|)|$\mu_{\tilde{\tilde{\tilde{|}}}}$)$_C$B,

$(|\tilde{\tilde{\tilde{|}}}|)|\mu_{\tilde{\tilde{\tilde{|}}}}{}^{q_n}_{q_1}\cdots$,

$(|\tilde{\tilde{\tilde{|}}}|)|\mu_{\tilde{\tilde{\tilde{|}}}}{}^{q_B}_{q_A}\cdots$,

$$(|\tilde{\tilde{|}}|)|\mu_{\tilde{|}\tilde{|}\tilde{|}}\begin{matrix}\{any\ C\}\\ \cdots\\ \{any\ D\}\end{matrix},$$

$$\text{fsel}\begin{matrix}q_n\\ \cdots\\ q_1\end{matrix}\text{f},$$

parelf(fsself – type),

parelf$((|\tilde{\tilde{|}}|)|\mu_{\tilde{|}\tilde{|}\tilde{|}})$,

parelf$\overset{fZ}{\longrightarrow}$ parelf, fZ is any fuzzy Z, Z is any,

parelf$\overset{fZ}{\longrightarrow}$singelf,

singelf$\overset{fZ}{\longrightarrow}$parelf,

singelf$\overset{fZ}{\longrightarrow}$singelf,

$\text{parel}_N((|\tilde{\tilde{|}}|)|\mu_{\tilde{|}\tilde{|}\tilde{|}})$,

$\text{parel}_N(\text{parel}_N((|\tilde{\tilde{|}}|)|\mu_{\tilde{|}\tilde{|}\tilde{|}}))$,

$\text{pasel}_N(\text{pa}(|\tilde{\tilde{|}}|)|\mu_{\tilde{|}\tilde{|}\tilde{|}})$,

$\text{sel}_N((|\tilde{\tilde{|}}|)|\mu_{\tilde{|}\tilde{|}\tilde{|}})$.

Remark 2. 2.3.1. May consider self-hierarchy of interpretations, self-type hierarchy of interpretations, $(|\tilde{\tilde{|}}|)|\mu_{\tilde{|}\tilde{|}\tilde{|}}$-type hierarchy of interpretations, Spiral$(|\tilde{\tilde{|}}|)|\mu_{\tilde{|}\tilde{|}\tilde{|}}$, Spiral pa $(|\tilde{\tilde{|}}|)|\mu_{\tilde{|}\tilde{|}\tilde{|}}$.

Remark 2. 2.3.2. A measure can be introduced for any dynamic operator.

Remark 2. 2.3.3. The actions of a short-pulse laser has the following structures: $(|\tilde{\tilde{|}}|)|\mu_{\tilde{|}\tilde{|}\tilde{|}}$fself$((|\tilde{\tilde{|}}|)|\mu_{\tilde{|}\tilde{|}\tilde{|}})$, ↑ | ↓$(|\tilde{\tilde{|}}|)|\mu_{\tilde{|}\tilde{|}\tilde{|}}$fself$(|||)$, ↑ | ↓$(|\tilde{\tilde{|}}|)|\mu_{\tilde{|}\tilde{|}\tilde{|}}$.

Ordered pafself

$\overrightarrow{pafself(d)} = d_x(\overrightarrow{(|\tilde{\tilde{|}}|)|\mu_{\tilde{|}\tilde{|}\tilde{|}}})(-d_y)$ (),

x, y are in particular, different space places or x =y.

Here d is in different space places: by the form d in x - d_x and the form (-d) - ($-d_y$) simultaneously. d may be a fuzzy program. Then $\overrightarrow{pafself(d)}$ will be ordered pafself program.

Ordered $(|\tilde{|}|)|\mu_{\tilde{|||}}$

$A\overrightarrow{(|\tilde{|}|)|\mu_{\tilde{|||}}}$ B is A $(|\tilde{|}|)|\mu_{\tilde{|||}}$ into B, but not vice versa (but not on the contrary).

Remark 2. 2.3.4. Codes for fself are 1, $(|\tilde{|}|)|\mu_{\tilde{|||}}$ - 0, and binary arithmetic can be used for programming. For example, fself + $(|\tilde{|}|)|\mu_{\tilde{|||}}$ + $fself^3$ + $((|\tilde{|}|)|\mu_{\tilde{|||}})^6$. For normal manipulation: fself + $fself^3$ +. For self- manipulation: $(|\tilde{|}|)|\mu_{\tilde{|||}}$ + $((|\tilde{|}|)|\mu_{\tilde{|||}})^6$. Nano- (space-time) by $((|\tilde{|}|)|\mu_{\tilde{|||}})^{-1}$, ${}^{q}_{q}$Sprt, Nano-elements by $((|\tilde{|}|)|\mu_{\tilde{|||}})^{-1}$, ${}^{q}_{q}$Sprt, Nano-any C by $((|\tilde{|}|)|\mu_{\tilde{|||}})^{-1}$, ${}^{q}_{q}$Sprt. Briefly pulsed laser as a light identifyer can nano-disidentify matter. Ultraviolet UVC radiation is closer to self, UHF is closer to paself.

Any object has a self-type, $(|\tilde{|}|)|\mu_{\tilde{|||}}$-type root, of which it is a manifestation. The CNS has its own self-type, $(|\tilde{|}|)|\mu_{\tilde{|||}}$-type root.

Remark 2. 2.3.5. The usual 2-interpretation comes from the nature of the mind, which is determined by the nature of the fuzzy internal dialogue, i.e., the activation of B (in particular, the cause) activates C (the effect). In the case of other interpretations (e.g., N-interpretations, N>2), e.g., through the will, all sorts of activations of various elements of the fuzzy interpretation fuzzy structure are possible, which is what this fuzzy interpretation will fuzzy consist of.

Remark 2. 2.3.6. Fuzzy Dynamic operators can be used as interpretations forms.

Let’s introduce Fd = probability|||definiteness.

Consider p||| = ${}^{|||}_{|||}$Sprt${}^{|||}_{|||}$.

Let’s introduce

Ffd = probability($\tilde{|||}$)|$\mu_{\tilde{|||}}$definiteness

Consider p($\tilde{|||}$)|$\mu_{\tilde{|||}}$

May consider the following notations: Singular operations (actions), Singular fuzziness, Singular space, Fuzzy singularities etc.

The ≡ sign was used as a connector to define |||- type and self- type singularities above. Here we will consider other types of singularities using other connector signs. To do this, we will introduce the following connectors/signs:

1) inequality-identification connectors-signs
   a) $\underline{\geq}$. Example: singularity H > 2H, ∀H, interpretation form – (1, (2<, 1)),
   b) $\underline{\underline{\geq}}$. Example: singularity H ≥ 2H, ∀H, interpretation form – (1, (2$\underline{<}$, 1)),
   c) $\underline{\underline{\leq}}$.
   d) $\underline{\underline{\underline{\leq}}}$.
   e) $\underline{\rightarrow}$. Examples: singularity A → B, ∀B, singularity A → B, ∀A,
   f) $\underline{\epsilon}$. Example: singularity 2H ϵ H, ∀H, interpretation form – (1, (2ϵ, 1)),
   g) $\underline{\underline{\supset}}$. Example: singularity H ⊃2H, ∀H, interpretation form – (1, (2⊂, 1)),
   h) $\underline{\subseteq}$. Example: singularity 2H ⊂ H, ∀H,
   i) c y b = q, ∀c, b, q; y is singular operation, action,
   j) Etc.

Remark 2. 2.3.6.1. The designation ϵ⊃ is ϵ|||⊃. May consider self(ϵ⊃), pself(ϵ⊃),paself(ϵ⊃) oself(ϵ⊃),parelf(ϵ⊃),singelf(ϵ⊃) etc.

Remark 2. 2.3.7. The central nervous system, under stress, “freezes” and switches the normal flow mode (physical space with time t) to the mode of living energy

fibers (energy space with different times depending on the “point of application”) with the center in the Will.

Remark 2. 2.3.8. The energy of non-living things can be imagined as a self from the energy of the elementary particles of which it consists. Self-type(for example, self, oself) and |||-type work with Energies only. The energy D of the bonds between the elements of an inanimate object (in particular, elementary particles, which are the result of $|||^{-1}$of this inanimate object) can act as its subtle body. ||| of this Energy acts as an element of its upper level, which corresponds to the self-structure of this inanimate object. If the energy of an inanimate object were not self-type, i.e., were not closed on itself, it would simply disintegrate, and the object itself would too. The binding energy of the elements of an inanimate object is: D = A|||A, where A is the elementary particle energy. A = D$|||^{-1}$D = $\frac{D}{D}$St = oself(D). If in the equations for D, instead of the = sign, we use the sign $|||^{-1}$ = $\overset{=}{=}$St = oself(=), then the resulting relations will form oself-type singularities for the D of $|||^{-1}$-level.

Remark 2. 2.3.9. For a living organism, DNA is the carrier of its "gross," i.e., material, part, while the embryonic "double" is the carrier of its "subtle" part.

Remark 2. 2.3.10. Any Energy is characterized by a level of hierarchy, degree of freedom, structure, and meaning at this level of hierarchy.

Remark 2. 2.3.11. In order to apply SmnSprt to the desired goal, task, object, action, process, it is necessary that the resources of SmnSprt are sufficient and then the energy shell of SmnSprt|||the energy shell of the desired one, which gives the energy of SmnSprt|||the energy of the desired one as a whole.

Remark. May consider interpretation A|||interpretation B, A, B are any. For example, 2- interpretation|||3-interpretation gives Dynamic science, in particular, Dynamic Mathematics.

2. 3 Fundamentally new approachs to physics

2. 3. 1 Fundamentally new approach to elementary particles

Definition 2. 3. 1. 1. The dynamic element $\overset{\overline{a}\,(t)}{g(t)}\ \mathrm{SCprt(t)}\ \overset{\overline{a}\,(t)}{g(t)}$ is the process of containing $\overline{a}\,(t)$ within itself as an element by $g(t)$ and the process of expelling $\overline{a}\,(t)$from itself as an element by $g(t)$ simultaneously [17].

Definition 2. 3. 1. 2. The dynamic element $\mathrm{SCprt(t)}\ \overset{\overline{a}\,(t)}{g(t)}$ is the process of containing $\overline{a}\,(t)$ within itself as an element by $g(t)$ [17].

Definition 2. 3. 1. 3. The dynamic element $\overset{\overline{a}\,(t)}{g(t)}\ \mathrm{SCprt(t)}$ is the process of expelling $\overline{a}\,(t)$ from itself as an element by $g(t)$ [17].

Definition 2. 3. 1. 4. The dynamic element $\overset{\{\,\}}{g(t)}\ \mathrm{SCprt(t)}$ is the process of expelling $\overline{a}\,(t)$ from the emptiness by $g(t)$ [17].

The Law of **Dark Matter**

$$\mathrm{oself}(\overline{a}\,(t)) = \overset{\overline{a}\,(t)}{g(t)}\ \mathrm{SCprt(t)} = |||^{-1}_{\overline{a}\,(t)}(\overline{a}\,(t)).$$

A more general ***Law*** $|||^{-1}$

$$|||^{-1}_{\left(\overline{b}\,(t),\ \overline{c}\,(t)\right)}(\overline{a}\,(t)) = \overline{b}\,(t),\ \overline{c}\,(t) \text{ for } \forall\ \overline{b}\,(t),\ \overline{c}\,(t),\ \overline{a}\,(t).$$

Energy $|||^{-1}$of emptiness gives birth to $\overline{a}\,(t)$. So, the production of new elementary particles seems to come from nowhere. Either we take from emptiness by $|||^{-1}$ or through substitution by $|||$.

Since our dynamic mathematics places the primary emphasis on the dynamics of energy, rather than on objectivity, the level of approach to studying processes expands.

Manifestations of the upper level, corresponding to elementary particle, are: 1) the object itself without self-interaction, 2) the spin of self-interaction, 3) other properties of both self-interaction and interactions etc.

Mathematically, the simplest interpretation of some elementary particle at its level pa||| and their characteristics will be represented as the following dynamic operator

$\overline{a}(t)$ $\overline{a}(t)$
$(action\ Q(t))^{-1}$Dprt(t) $action\ Q(t)$ [2], $\overline{a}(t) = \overline{a}_0(t) + E(t)$, pa||| = $|||^{-1}(|||)|||$, with
$\overline{a}(t)$ $\overline{a}(t)$

manifestations in the our usual level: mass $\| \overline{a}_0(t) \|$ and spin $\| \overline{a}_0(t) \|_{\rightarrow}$ etc.

Mathematically, an electron at its level pa||| will be represented as the following

$\overline{f}(t)$ { }
dynamic operator $(action\ Q(t))^{-1}$Dprt(t) { }, { } is the emptiness, if $action\ Q(t)$ is
$\overline{f}(t)$ { }

$\overline{c}(t)$ { }
the containment by type $g(t)$, then as $g(t)$ SCprt(t) { }, foton as
$\overline{c}(t)$ { }

$\overline{a}(t)$ { } $\overline{a}(t)$ { }
$(action\ Q(t))^{-1}$Dprt(t) { } $(action\ Q(t))^{-1}$Dprt(t) { }
$\overline{a}(t)$ { } $\overline{a}(t)$ { }
($action\ Q(t))^{-1}$ Dprt(t) ($action\ Q(t))^{-1}$ or
$\overline{a}(t)$ { } $\overline{a}(t)$ { }
$(action\ Q(t))^{-1}$Dprt(t) { } $(action\ Q(t))^{-1}$Dprt(t) { }
$\overline{a}(t)$ { } $\overline{a}(t)$ { }

$\overline{a}(t)$ { } $\overline{a}(t)$ { }
$g(t)$ SCprt(t) { } $g(t)$ SCprt(t) { }
$\overline{a}(t)$ { } $\overline{a}(t)$ { }
$g(t)$ Sprt(t) $g(t)$,
$\overline{a}(t)$ { } $\overline{a}(t)$ { }
$g(t)$ SCprt(t) { } $g(t)$ SCprt(t) { }
$\overline{a}(t)$ { } $\overline{a}(t)$ { }

Higgs boson as

$\overline{b}(t)$ { } $\overline{a}(t)$ { }
$g(t)$ SCprt(t) { } $g(t)$ SCprt(t) { }
$\overline{b}(t)$ { } $\overline{a}(t)$ { } { } $\overline{p}(t)$
$g(t)$ Sprt(t) $g(t)$, proton is { } SCprt(t) $g(t)$ =
$\overline{a}(t)$ { } $\overline{b}(t)$ { } { } $\overline{p}(t)$
$g(t)$ SCprt(t) { } $g(t)$ SCprt(t) { }
$\overline{a}(t)$ { } $\overline{b}(t)$ { }

{ } $u(t)$ $g_1(t)$ $u(t)$
{ } { } PrSCprt $g_2(t)$ $g_2(t)$,
{ } { } { } $d(t)$

{ } $\overline{n}(t)$ { } { } { } $u(t)$
neutron as { } SCprt(t) $g(t)$ = { } { } PrSCprt $g_3(t)$ $g_3(t)$, quarks d(t) =
{ } $\overline{n}(t)$ { } $d(t)$ $g_4(t)$ $d(t)$

{ } $\overline{r}(t)$) { } $\overline{d}(t)$
{ } SCprt(t) $g(t)$, u(t) = { } SCprt(t) $g(t)$ interact with each other by gluon fields
{ } $\overline{r}(t)$ { } $\overline{d}(t)$

$\overline{h_j}(t)$ $\overline{v_j}(t)$
$g_j(t) = w_j(t)$SCprt(t) $w(t)$, j = 1, 2, 3, 4, which are manifestations of general gluon
$\overline{h_j}(t)$ $\overline{v_j}(t)$

$\overline{h}(t)$ $\overline{v}(t)$
$w(t)$ SCprt(t) $w(t)$
$\overline{h}(t)$ $\overline{v}(t)$

field in the areas between certain quarks, graviton as $w(t)$ SCprt(t)

$\overline{h}(t)$ $\overline{v}(t)$
$w(t)$ SCprt(t) $w(t)$
$\overline{h}(t)$ $\overline{v}(t)$

$\overline{h}(t)$ $\overline{v}(t)$
$w(t)$ SCprt(t) $w(t)$
$\overline{h}(t)$ $\overline{v}(t)$

$w(t)$ etc. Gluons with quarks demonstrate energy closed in on

$\overline{h}(t)$ $\overline{v}(t)$
$w(t)$ SCprt(t) $w(t)$
$\overline{h}(t)$ $\overline{v}(t)$

itself (energy "in itself"). Orbitals also clearly demonstrate energy closed in on itself (energy "in itself").

Remark 0. Energy is connections. All connections… connections as internal and external of $\overline{a}(t)$ belong to self($\overline{a}(t)$) = $\overline{a}(t)$ $\overline{a}(t)$
$g(t)$ SCprt $g(t)$, in this, in particular, the
$\overline{a}(t)$ $\overline{a}(t)$

closedness of energy of type self($\overline{a}(t)$) in itself is manifested. The set of all connections… connections as internal and external of $\overline{a}(t)$ will be manifestations self-energy into our usual level.

Energy $\overline{a}(t)$ $\overline{a}(t)$
$g(t)$ SCprt $g(t)$ has 8 possibilities of transformation:
$\overline{a}(t)$ $\overline{a}(t)$

1) $\overline{a}(t)$ $\overline{a}(t)$ $\overline{a}(t)$ $\overline{b}(t)$ 2) $\overline{a}(t)$ $\overline{a}(t)$
$g(t)$ SCprt(t) $g(t)$ transform to $g(t)$ SCprt(t) $g(t)$, $g(t)$ SCprt(t) $g(t)$
$\overline{a}(t)$ $\overline{a}(t)$ $\overline{a}(t)$ $\overline{a}(t)$ $\overline{a}(t)$ $\overline{a}(t)$

2) transform to $g(t)^{\overline{a}(t)}$ SCprt(t) $g(t)^{\overline{b}(t)}$, 3) $g(t)^{\overline{a}(t)}$ SCprt(t) $g(t)^{\overline{a}(t)}$ transform to $g(t)^{\overline{a}(t)}$ SCprt(t) $g(t)^{\overline{b}(t)}$, 4) $g(t)^{\overline{b}(t)}$ SCprt(t) $g(t)^{\overline{a}(t)}$ transform to $g(t)^{\overline{b}(t)}$ SCprt(t) $g(t)^{\overline{b}(t)}$, 5) $g(t)^{\overline{a}(t)}$ SCprt(t) $g(t)^{\overline{a}(t)}$ transform to $g(t)^{\overline{a}(t)}$ SCprt(t) $g(t)^{\overline{a}(t)}$, 6) $g(t)^{\overline{a}(t)}$ SCprt(t) $g(t)^{\overline{a}(t)}$ transform to $g(t)^{\overline{b}(t)}$ SCprt(t) $g(t)^{\overline{b}(t)}$, 7) $g(t)^{\overline{a}(t)}$ SCprt(t) $g(t)^{\overline{a}(t)}$ transform to $g(t)^{\overline{a}(t)}$ SCprt(t) $g(t)^{\overline{b}(t)}$, 8) $g(t)^{\overline{a}(t)}$ SCprt(t) $g(t)^{\overline{a}(t)}$ transform to $g(t)^{\overline{b}(t)}$ SCprt(t) $g(t)^{\overline{b}(t)}$. The same for energy $(action\ Q(t))^{-1}$Dprt(t) $action\ Q(t)^{\overline{a}(t)}$.

Remark 2. 3. 1.1. Also, may consider operator $(action\ Q(t))^{-1}$ SDS(t) $action\ Q(t)^{\overline{a}(t)}$ [2] instead of operator $g(t)^{\overline{a}(t)}$ SCprt $g(t)^{\overline{a}(t)}$ for interpretation with elementary particles, in particular, $Subject\ of\ Q(t)$ may be an observer.

Spin self-interaction by $g(t)$ from $g(t)^{\overline{a}(t)}$ SCprt(t) $g(t)^{\overline{a}(t)}$: one-sided is half-integer, for example, $g(t)^{\overline{a}(t)}$ SCprt(t) $\{\}^{\overline{a}(t)}$, two-sided is integer, for example, $g(t)^{\overline{a}(t)}$ SCprt(t) $g(t)^{\overline{a}(t)}$.

Self- level corresponds to self-energies.

Entanglements in quantum mechanics

$\overline{a}\,(t)$ { }
$g(t)$ SCprt(t) { }
$\overline{a}\,(t)$ { }
$\overline{b}\,(t)$ { } SCprt(t), or $pSTp_1Rself(A)$, A is structure with elementary
$g(t)$ SCprt(t) { }
$\overline{b}\,(t)$ { }
particles.

Remark 2. 3.1.2. The observer, through his observation, manifests an elementary particle to the ordinary level and thereby works with its manifestation, i.e., as an ordinary-level object — a particle. Corresponding changes occur at the object's upper level.

Remark 2. 3. 1.2.1. We can only work with equations at the ordinary level; other levels can only correspond to singularity relations.

Remark 2. 3.1.3. Let' s consider different singularities for ensemble of elementary particles from relations:

1. $\frac{\breve{d}\breve{q_s}}{\breve{d}t} \cong [\breve{H}, \breve{q_s}\breve{]},$

$\frac{\breve{d}\breve{p_s}}{\breve{d}t} \cong [\breve{H}, \breve{p_s}\breve{]},$

where $\breve{q_s}$ is oself(q_s), $\breve{p_s}$ is oself(p_s), $\breve{H}$ is oself(H), $\cong$ is oself(=), [, $\breve{]}$ is oself([,]).

2. $\frac{\breve{d}\overline{\breve{q_s}}}{\breve{d}t} \overline{\cong} \ [\overline{\breve{H}}, \overline{\breve{q_s}}\breve{]},$

$\frac{\breve{d}\overline{\breve{p_s}}}{\breve{d}t} \overline{\cong} [\overline{\breve{H}}, \overline{\breve{p_s}}\breve{]},$

where $\overline{\breve{q_s}}$ is pself(q_s), $\overline{\breve{p_s}}$ is pself(p_s), $\overline{\breve{H}}$ is pself(H), $\overline{\cong}$ is pself(=), [, $\overline{\breve{]}}$ is pself([,]).

3. $\frac{\breve{d}\overline{\overline{\breve{q_s}}}}{\breve{d}t} \overline{\overline{\cong}} [\overline{\overline{\breve{H}}}, \overline{\overline{\breve{q_s}}}\breve{]},$

$\frac{\breve{d}\overline{\overline{\breve{p_s}}}}{\breve{d}t} \overline{\overline{\cong}} [\overline{\overline{\breve{H}}}, \overline{\overline{\breve{p_s}}}\breve{]},$

where $\overline{\overline{\breve{q}_s}}$is paself($q_s$),$\overline{\overline{\breve{p}_s}}$ is paself(p_s), $\overline{\overline{\breve{H}}}$ is paself(H), $\overline{\overline{\cong}}$ is paself(=), [,] is paself([,]).

4. $\frac{\breve{d}\overline{\overline{p_s}}}{\breve{d}t} \overline{\overline{\cong}}$ [$\overline{\overline{\breve{H}}}$, $\overline{\overline{\breve{p}_s}}$].

Remark 2. 3.1.4. Let' s consider different singularities for elementary particles

|||Schrödinger equation

|||(iħ∂ψ/∂t = $\hat{H}$ψ)

|||Dirac equation

|||(− $\frac{\hbar}{i}$ i $\frac{\partial}{\partial t}$ Ψ − $\frac{c\hbar}{i}$ (α · ∇)Ψ − $c^2\rho_3\Psi$ = 0).

Remark 2. 3.1.5. Probability is $|||^{-1}$-characteristic, that is, instead of one characteristic, we have a series of values in the form of a distribution of its probabilities. May consider $pSTp_1Rself(A)$, A is a probability.

Levels of $|||^{-1}$:

1) To molecules
2) To atoms
3) To other elementary particles

Remark 2. 3.1.6. Quantum logic is an example of $|||^{-1}$- logic.

Remark 2. 3.1.7. Two-way time in quantum mechanics $\begin{matrix} \textit{time to the future} \\ \rightarrow \\ \leftarrow \\ \textit{time into the past} \end{matrix}$ -wave is oself(wave).

Remark 2. 3.1.8. May consider oself-point of space, pself-point of space, paself-point of space, parelf-point of space, singelf-point of space etc.

2. 3. 2 Induction

The analogue of Maxwell equations for any induction:

$$\mathrm{rot}(Q) = -(1/q_0)\,\partial(R)\ /\partial t,\ (*_H)$$
$$\mathrm{rot}(S) = j/q_0 + (1/q_0)\cdot\partial\,(W)\ /\partial t,\ (**_H)$$

Q – tension of field of action G, R – induction of induction field for Q, S - tension of induction field for Q, W – induction of field for Q, where j, q_0 are the corresponding coefficients.

Remark 2. 3. 2.1. Entanglements in quantum mechanics here correspond to higher than usual levels of hierarchy. For example, self-level, which allows define spin of elementary particles, in particular.

Induction law(Generalization of Newton's third law)

Change of magnitude of A on usual level: ΔA in the in a medium B leads to (A +ΔA)|||B in the upper level, the reaction of which manifests itself at a lower level as $\mathrm{self}_B^{3/2}(\Delta A)$, which is a type of self-combining and is usually realized by a vortex, depending on the medium B.

Remark 2. 3. 2.2. The hierarchy of layers in the human cocoon corresponds to a portion of the hierarchy of energy fibers in the Universe. Naturally, any change in one layer causes a corresponding induction in adjacent layers.

Remark 2. 3.2. 2. The stopping of the flow of the Universe transfers to the upper layer (layer of |||) of the hierarchy, which leads to the effects of attosecond physics. At the upper level ||| there is no time—there everything is one; at the level $|||^{-1}$, associated with elementary particles, there is no time—there everything is separate.

Law of Return (Repetition)

Return (repetition) of A, A is any, is actually the conservation of energy of A. Symmetry is a mirror repetition.

May consider the following variants of symmetry: Sprt $\begin{matrix}\text{symmetry} & \text{symmetry}\\ \text{symmetry'} & \text{symmetry}\end{matrix}$ Sprt, $\begin{matrix}\text{symmetry}\\ \text{symmetry}\end{matrix}$Sprt $\begin{matrix}\text{symmetry}\\ \text{symmetry}\end{matrix}$ etc.

Repetition of B is conservation of B and force, associated with this. Self is one of its extreme forms. Repetition of B create “hole” energy.

The next level of repetition is a wave B. The next level of repetition is Self(B) etc.

May consider oself- symmetry of space, pself- symmetry of space, paself- symmetry of space, parelf- symmetry of space, singelf- symmetry of space,

oself- repetition of space, pself- repetition of space, paself- repetition of space, parelf- repetition of space, singelf- repetition of space.

May consider the following variants of repetition: Sprt $\begin{matrix}\text{repetition} & \text{repetition}\\ \text{repetition'} & \text{repetition}\end{matrix}$Sprt, $\begin{matrix}\text{repetition}\\ \text{repetition}\end{matrix}$Sprt $\begin{matrix}\text{repetition}\\ \text{repetition}\end{matrix}$ etc.

2.4 Elements of Dynamic Biology

Will is a self from the assemblage point in position. Will of the magician is paself. Will of the magician can |||, (for example, ||| for A and B: A↑ I ↓B), and can pa|||. Will of the magician is Intention. Management of Will is carried out by Intention. Creation of SmnSprt is planned on these principles.

Some analogy between the structures of living and non-living things: the electron orbitals of an atom correspond to the meridians on the skin of a living organism.

$$\left(\begin{array}{c} \dots \\ Manager\ of\ Intention(parelf) \\ Intention(pa|||) \\ Will(paself) \\ ||| \\ Creation\ of\ self(self^{\frac{3}{2}}) \\ self \\ \dots \end{array}\right)(*)$$

Remark 2. 4.0. The absence of uricase metabolism in humans, unlike other mammals, allows for the inclusion of "internal dialogue" and then abstract thinking.

Remark 2. 4.1. Entanglements in quantum mechanics here correspond to higher than usual levels of hierarchy. For example, self-level, which allows define spin of elementary particles, in particular.

. The isolation of an object (i.e. $|||^{-1}$) in the energy space corresponds to its energic nature $pa||| = |||(|||)|||^{-1}$.

Remark 2. 4.2. As point of Energetic space may try to take Vprt element [20].

Remark 2. 4.3. Self-type(for example, self, oself) and |||-type work with Energies only.

The energy D of the bonds between the elements of an inanimate object (in particular, elementary particles, which are the result of $|||^{-1}$of this inanimate object) can act as its subtle body. ||| of this Energy acts as an element of its upper level, which corresponds to the self-structure of this inanimate object.

For a living organism, DNA is the carrier of its "gross," i.e., material, part, while the embryonic "double" is the carrier of its "subtle" part. $Sprt^{DNA}_{DNA}$ – closing on itself (ends) before division.

Any Energy is characterized by a level of hierarchy, degree of freedom, structure, and meaning at this level of hierarchy.

Remark 2. 4.4. Towards the Designing Pseudo-Living Energy: upon receipt of funding, it is planned the creation of Energy fibers of any structure through nano-self-technologies and Creation of any structure from their bases. Similarly to nano-self-technological actions.

Remark 2. 4.5. Interpretations of reason (the cortex (surface) of the brain) in words and numbers, Interpretations of Will (subcortex and from the center of the body) in actions (processes) as well as SmnSprt.

Remark 2. 4.6. Our Dynamic Science has thrown a "bridge" to magic: from ordinary physical actions to energetic actions.

Remark 2. 4.7. Dynamic Operators with biological (Dynamic biology) and chemical (Dynamic chemistry) actions, etc. to energetic actions, in particular, through degenerations.

Remark 2. 4.8. Physical space with objects is 1 point of energy space by human energic cocoon. Ordinary people are "caught" by the limitations of physical space, magicians - by the limitations of energy space. Energy space-time is Not just a flow, but any possible structure.

Let’s consider the following definitions:

Definition 2. 4.9. The partial set A is set A = (B, C, D), where A is defined, B is undefined, but it can be defined through A, C cannot be determined at all.

Definition 2. 4. 10. The partial space is space A = (B, C, D), where A is defined, B is undefined, but it can be defined through A, C cannot be determined at all.

Let’s consider energetic space A corresponding to Definition 2 and it’s part B. C cannot be defined at all. Let our interpretation as the Universe be defined as one of the possible choices from A through the entanglement of the interpretations of all normal organic beings through the position of the assemblage point at a given moment in time, which is also an interpretation in one of the parts of A. This is the

part in which the elementary particles of our world exist. Other structures exist in other parts of A.

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 «ΛΌΓΟΣ» 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 Pasynkov 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: 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
4) Danilishyn I., Danilishyn O. DYNAMIC SETS THEORY: SIT-ELEMENTS AND THEIR APPLICATIONS. Preprint. Research Square. *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

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 WISSENCHAFTLICHEN 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 «ΛΌΓΟΣ» 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 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

15) 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. Introduction to Dynamic Operators: Lprt-Elements and Applications to Physics and Other their Applications. https://www.opastpublishers.com/journal/journal-of-mathematical-techniques-and-computational-mathematics/articles-in-press

16) Illia Danilishyn, Oleksandr Danilishyn. Introduction to Dynamic Mathematics: Zprt-elements and Their Applications. American Journal of Mathematical and Computer Applications, Volume 1 | Issue 1 (2025), pp.1-146. https://cskscientificpress.com/articles_file/278-_article1742642034.pdf

17) Illia Danilishyn, Oleksandr Danilishyn. Mathematical fundamentals of constructing pseudoliving energy theory and dynamic programmings: monograph / ed. by Pasynkov V. – Shawnee, USA: Primedia eLaunch LLC. – 2025. – 382 p. ISBN 979-8-89660-275-0 https://doi.org/10.36074/mfocp-letadp.monograph-2025
https://books.google.com.ua/books/about?id=j1xSEQAAQBAJ&redir_esc=y
https://explore.openaire.eu/search/publication?pid=10.36074%2Fmfocp-letadp.monograph-2025
https://dynamical-math.com/materials.html

18) Illia Danilishyn, Oleksandr Danilishyn. Mathematical uncertainties and their applications: monograph / ed. by Pasynkov V. – Shawnee, USA : Primedia eLaunch LLC. – 2025. – 612 p. ISBN 979-8-89660-284-2. DOI: https://doi.org/10.36074/muata.monograph-2025
https://dynamical-math.com/materials.html

19) Illia Danilishyn and Oleksandr Danilishyn. Elements of Dynamic Science. Journal of Modern Classical Physics & Quantum Neuroscience. Vol:1,3. Pg:1-35. DOI: doi.org/10.63721/25JPQN0112.
https://wmjournals.com/physics.html

20) I. Danilishyn, & O. Danilishyn, (2025). Elements of dynamic science: monograph. *Primedia ELaunch LLC*, 573. https://doi.org/10.36074/eods.monograph-2025

21) Galushkin A. Networks: principles of the theory. Hot line-Telecom. M.,2010 (in Russian).

22) Illia Danilishyn, Oleksandr Danilishyn. Introduction to Dynamic Mathematics: Zprt-elements and Their Applications. American Journal of Mathematical and Computer Applications, Volume 1 | Issue 1 (2025), pp.1-146. https://cskscientificpress.com/articles_file/278-_article1742642034.pdf

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 «ΛΌΓΟΣ» 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. *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

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 WISSENCHAFTLICHEN 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 «ΛΌΓΟΣ» 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 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

15) 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. Introduction to Dynamic Operators: Lprt-Elements and Applications to Physics and Other their Applications. https://www.opastpublishers.com/journal/journal-of-mathematical-techniques-and-computational-mathematics/articles-in-press

16) Illia Danilishyn, Oleksandr Danilishyn. Introduction to Dynamic Mathematics: Zprt-elements and Their Applications. American Journal of

Mathematical and Computer Applications, Volume 1 | Issue 1 (2025), pp.1- 146. https://cskscientificpress.com/articles_file/278-_article1742642034.pdf

17) Illia Danilishyn, Oleksandr Danilishyn. Mathematical fundamentals of constructing pseudoliving energy theory and dynamic programmings: monograph / ed. by Pasynkov V. – Shawnee, USA: Primedia eLaunch LLC. – 2025. – 382 p. ISBN 979-8-89660-275-0 https://doi.org/10.36074/mfocp-letadp.monograph-2025 https://books.google.com.ua/books/about?id=j1xSEQAAQBAJ&redir_esc=y https://explore.openaire.eu/search/publication?pid=10.36074%2Fmfocp-letadp.monograph-2025 . https://dynamical-math.com/materials.html

18) Illia Danilishyn, Oleksandr Danilishyn. Mathematical uncertainties and their applications: monograph / ed. by Pasynkov V. – Shawnee, USA : Primedia eLaunch LLC. – 2025. – 612 p. ISBN 979-8-89660-284-2. DOI: https://doi.org/10.36074/muata.monograph-2025 https://dynamical-math.com/materials.html

19) Illia Danilishyn and Oleksandr Danilishyn. Elements of Dynamic Science. Journal of Modern Classical Physics & Quantum Neuroscience. Vol:1,3. Pg:1-35. DOI: doi.org/10.63721/25JPQN0112. https://wmjournals.com/physics.html

20) I. Danilishyn, & O. Danilishyn, (2025). Elements of dynamic science: monograph. *Primedia ELaunch LLC*, 573. https://doi.org/10.36074/eods.monograph-2025

21) Illia Danilishyn, Oleksandr Danilishyn. Introduction to Dynamic Mathematics: Zprt-elements and Their Applications. American Journal of Mathematical and Computer Applications, Volume 1 | Issue 1 (2025), pp.1- 146. https://cskscientificpress.com/articles_file/278-_article1742642034.pdf

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 III. Mathematical fundamentals of the Dynamic programming by pself-type and paself-type

Introduction

Dynamic programming by pself-type and paself-type works through levels hierarchy in the space of energies with pseudo-living 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. :≡ (recurrent identity) is the dynamic assignment (recurrent assignment (when subsequent values of variables are pa||| through the previous ones)). Differs from the usual software environment and pa||| 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 paself-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 paself-type structures, at the upper ones - with all singularities and with itself in all possible upper modes (paself-development, paself-clotting). paself-development of Dynamic programming can be, for example, in the presence of its "embryo", for example, in the structure $\overset{x}{\{\text{Dynamic programming ,its "embryo"}\}}Sprt_{\text{x}}^{\{\text{Dynamic programming ,its "embryo"}\}}$. Then from this "embryo" it can grow a virtual double with much greater capabilities. For example, to replace A with B, going to the self-level: $A|||A^{-1}$, we go down to the lower level with A|||B, then, emphasizing B and going down to the next lower level, we get B. In order to apply SmnSprt to the desired goal, task, object, action, process, it is necessary that the resources of SmnSprt are sufficient

and then the energy shell of SmnSprt|||the energy shell of the desired one, which gives the energy of SmnSprt|||the energy of the desired one as a whole.

3.1 Program operators Sprt and their pself-type

Introduction

There is a need to develop an instrumental mathematical base for new technologies. An example of a parallel program and paself-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, 15-20].

3.1.1 Program operators Sprt and their pself-type

For simultaneous *solution of Q by D* and multiplication:

$\underset{solution\ of\ Q\ by\ D}{^{task\ Q}}Sprt_{\mathrm{x}}^{\{a*\}}$: the notation of the set B with elements $b_{i_1 i_2 \ldots i_n} = (\underset{solution\ of\ Q\ by\ D}{^{task\ Q}}Sprt_{x}^{\left\{a_{1_{i_1}}*, a_{2_{i_2}}*, \ldots,\ a_{n_{i_n}}\right\}})_R$ for any $\{i_1, i_2, \ldots, i_n\}$, x = $Sprt_w^{\{K\}}$ without repetitions, K-set of any $\{k_1*, k_2*, \ldots, k_n*\}$ without repeating them, k_i-any digit, i=1,2,…,n, R= $Sprt_w^{\{i_1+, i_2+, \ldots, i_n\}}$, R is the index of the lower discharge (we choose an index on the scale of discharges): see Table I.1.

Then $Sprt_{\mathrm{x}}^{\{B+\}}$ 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 Sprt-multiplication:

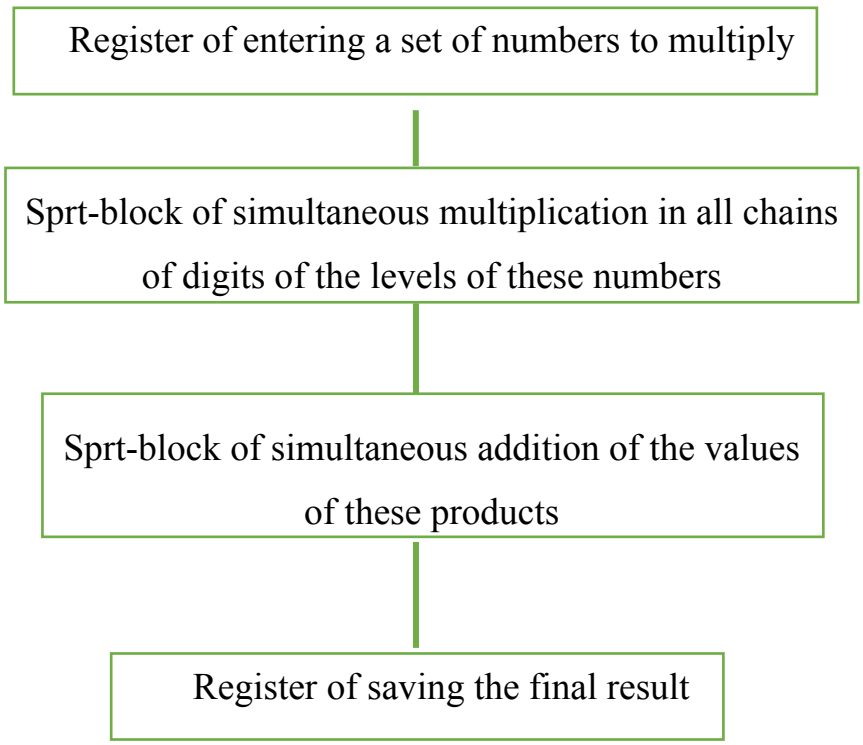


Fig. 3.1. The straightforward functional scheme of the assumed arithmetic-logical device for Sprt-multiplication.

Remark 3.1.1. 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.

Remark 3.1.1.1. Similarly simultaneous *solution of Q by D* and multiplication is implemented for ordered case: $\begin{matrix} task\ Q \\ solution\ of\ Q\ by\ D \end{matrix} \overrightarrow{Sprt}_{\mathsf{x}}^{\{a*\}}$.

Here are some of the Sprt programming operators [2]:

1. Assignment of the expressions $\{p\} = (p_1,p_2,...,p_n)$ to the variables $\{\mathsf{a}\} = (\mathsf{a}_1,\mathsf{a}_2,...,\mathsf{a}_n)$ and expelling of the assignment of expressions $\{r\} =$

$(r_1,r_2,...,r_n)$ from the variables $\{b\} = (b_1,b_2,...,b_n)$ simultaneously. This is implemented via $\{\{b\} \overset{x}{:=} \{r\}\}^{Sprt_{\mathsf{x}}^{\{\{\mathsf{a}\}:=\{p\}\}}}$ or $\{\{b\} \overset{x}{:=} \{r\}\}^{\overrightarrow{Sprt}_{\mathsf{x}}^{\{\{\mathsf{a}\}:=\{p\}\}}}$ for ordered case.

2. Simultaneous p-checking the set of conditions $\{f\} = (f_1, f_2, ...,f_n)$ for the set of expressions$\{B\} = (B_1, B_2, ...,B_n)$. Implemented via $IF\{\{B\}\overset{x}{\{f\}}\}\ then\ Q^{Sprt_{\mathsf{x}}^{IF\{\{B\}\{f\}\}\ then\ Q}}$ or $IF\{\{B\}\overset{x}{\{f\}}\}\ then\ Q^{\overrightarrow{Sprt}_{\mathsf{x}}^{IF\{\{B\}\{f\}\}\ then\ Q}}$ for ordered case, where Q can be anything, in the first here is expelling then containment.
3. Similarly for loop operators and others.

Analogue of $S_3 f$ for paself– software operators will differ only in that the aggregates $\{\mathsf{a}\},\{p\},\{B\},\{f\}$ will be formed from the corresponding Sprt program operators in analogue of form (1.1) and for more complex operators in the form from the analogues of forms (1.1.1) – (1.4) for pself.

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 analogue of $S_3 f$ for pself can be used for programming [5]:

1. The process of assignment of the expressions $\{p(t)\} = (p_1(t),p_2(t),...,p_n(t))$ to the variables $\{g(t)\} = (g_1(t),g_2(t),...,g_n(t))$ and the process of expelling of assignment of the expressions $\{r(t)\} = (r_1(t),r_2(t),...,r_n(t))$ from the variables $\{w(t)\} = (w_1(t),w_2(t),...,w_n(t))$ simultaneously is implemented through $\{\{w(t)\} \overset{x}{:=} \{r(t)\}\}^{Sprt(t)_{\mathsf{x}}^{\{\{g(t)\}:=\{p(t)\}\}}}$ or $\{\{w(t)\} \overset{x}{:=} \{r(t)\}\}^{\overrightarrow{Sprt}(t)_{\mathsf{x}}^{\{\{g(t)\}:=\{p(t)\}\}}}$ for ordered case.

2. The process of simultaneous p-check the set of conditions $\{f(t)\} = (f(t)_1, f_2,(t) ...,f(t)_n)$ for a set of expressions$\{B(t)\} = (B_1(t), B_2(t), ...,B_n(t))$ is implemented through$IF\{\{B(t)\}\overset{x}{\{f(t)\}}\}\ then\ Q(t)^{Sprt(t)_{\mathsf{x}}^{IF\{\{B(t)\}\{f(t)\}\}\ then\ Q(t)}}$ or

$IF\{\{B(t)\}\{\overset{x}{f}(t)\}\}\ then\ Q(t)$ $\overrightarrow{Sprt}(t)_{\mathsf{x}}^{IF\{\{B(t)\}\{f(t)\}\}\ then\ Q(t)}$ for ordered case, where Q(t) can be any.

3.Similarly for loop operators and others.

Analogue of $S_3 f(t)$ for pself– software operators will differ only in that the aggregates $\{\mathsf{a}\},\{p\},\{B(t)\},\{f(t)\}$ will be formed from corresponding processes Sprt(t) for the above-mentioned programming operators through analogue of form (1.1) for paself and for more complex operators in the form from the analogues of forms (1.1.1) – (1.4) for pself. We can consider the concept of a continual Sprt - element as ${}_{A}^{B}Sprt_{B}^{A}$, where A fits in continual capacity B. Then ${}_{B}^{B}Sprt_{B}^{B}$ will mean paself(B).

These elements are used for Sprt-coding, Sprt translation, coding pself, and translation pself for networks], which is suitable for electric current of ultrahigh frequency or by a short-pulse laser pulse.

3.1.2 The usage of Sprt-elements for networks

A. Galushkin's comprehensive monograph [21] 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 paself-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, paself-coding and paself-translation will be more stable. The target weights f_i in $\overset{a}{\{fx\}}Sprt_a^{\{fx\}}$ 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 $-\mathrm{sgn}(\overset{b}{\{ax\}}Sprt_b^{\{ax\}})$, b- mnSprt, $x = (x_1, x_2, \ldots, x_n)$ – source signals values, $a=(a_1, a_2, \ldots, a_n)$ – Sprt-synapses weights. The first level of mnSprt consists of simple mnSprt. The second level of mnSprt consists of $\overset{D}{\{mnSprt}Sprt_D^{\{mnSprt\}}$ – Sprt-node of mnSprt in range D, D- capacity for mnSprt node. The third level of mnSprt consists of $\overset{D}{\{St_D^{\{mnSt\}}\}}Sprt_D^{\{St_D^{\{mnSt\}}\}}$ - $Sprt^2$- node of mnSprt in range D, thus D becomes paself capacity for mnSprt. For our networks, it is sufficient to use $Sprt^2$- nodes of mnSprt, but paself-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 or by a short-pulse laser pulse. 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 -- $\overset{b}{g_2}\ \mathrm{SCprt}\ \overset{\{ax\}}{g_1}$, b- artificial neurons of $\{qy\}$ b

type SCprt (designation - mnSt), $x= (x_1, x_2, \ldots, x_n)$ are the values of the initial signals,, $a=(a_1,a_2,\ldots,a_n)$ are the weights of SCprt-synapses and the values of the output signals .

Remark 3.2.1. 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 $\underset{\{\text{eprograms}\}}{\overset{\text{mnSprt}}{}}Sprt_{\text{mnSprt}}^{\{\text{eprograms}\}}$, peprogram –executing program in Sprt-OS. Sprt-OS (or paself-OS) is based on Sprt-assembly language (or paself-assembly language), which is based on assembly language through Sprt-approach in turn, if the base of elements of Sprt-networks is sufficient. The peprograms are in Sprt-programming environments (or paself-programming environments), but this question and Sprt-networks base will be considered in the following monographs. In particular, peprograms may contain Sprt- programming operators. In mnSprt cores, the constant memory Sprt with correspondent peprograms 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 $\underset{\text{Smnsprt}}{\overset{\text{Smnsprt}}{}}Sprt_{\text{Smnsprt}}^{\text{Smnsprt}}$ 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 Sprt-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 peprograms 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 peprograms 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 peprogram if one is present of needed «target weight» are taken in target block at Sprt – activation of the networks. $\overset{activation}{\text{SmnSprt},f}Sprt_{activation}^{\text{SmnSprt},f}$ derives SmnSprt to the paself-level boundary with target weight f.

It’s used an alternating current of above high frequency and ultra-short optical pulses laser, 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.”

3.2 Program operators ffSprt and their pself-type

1 Similarly, for simultaneous multiplication $\underset{solution\ of\ Q\ by\ D}{\overset{task\ Q}{\mu}}\ \text{ffSprt}\ \underset{q}{\mu}$: the $\tilde{x}*$

notation of the fuzzy set B with elements $b_{i_1 i_2 \ldots i_n} =$

$$\left(\underset{solution\ of\ Q\ by\ D}{\overset{task\ Q}{\mu}}\ \text{ffSprt}^{\{x_{1_{i_1}} * \mu_{\tilde{x}}\left(x_{1_{i_1}}\right) * , x_{2_{i_2}} * \underset{q}{\mu}_{\tilde{x}}\left(x_{2_{i_2}}\right) * , \ldots, \ x_{n_{i_n}} * \mu_{\tilde{x}}\left(x_{n_{i_n}}\right)\}}\right)_R$$

for any $\{i_1, i_2, \ldots, i_n\}$ without repetitions, $q = \underset{\{K\}}{w} St_w^{\{K\}}$, K-set of any $\{k_1 *, k_2 *, \ldots, k_n *\}$ without repeating them, k_i-any digit, i=1,2,…,n, $R = \underset{\{i_1 +, i_2 +, \ldots, i_n\}}{w} St_w^{\{i_1+, i_2+, \ldots, i_n\}}$, R is the index of the lower discharge (we choose an index on the scale of discharges): see Table I.1.

Then $\underset{solution\ of\ Q\ by\ D}{\overset{task\ Q}{\mu}}\ \text{ffSprt}\ \underset{q}{\overset{B+}{\mu}}$ gives the final result of simultaneous p-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:

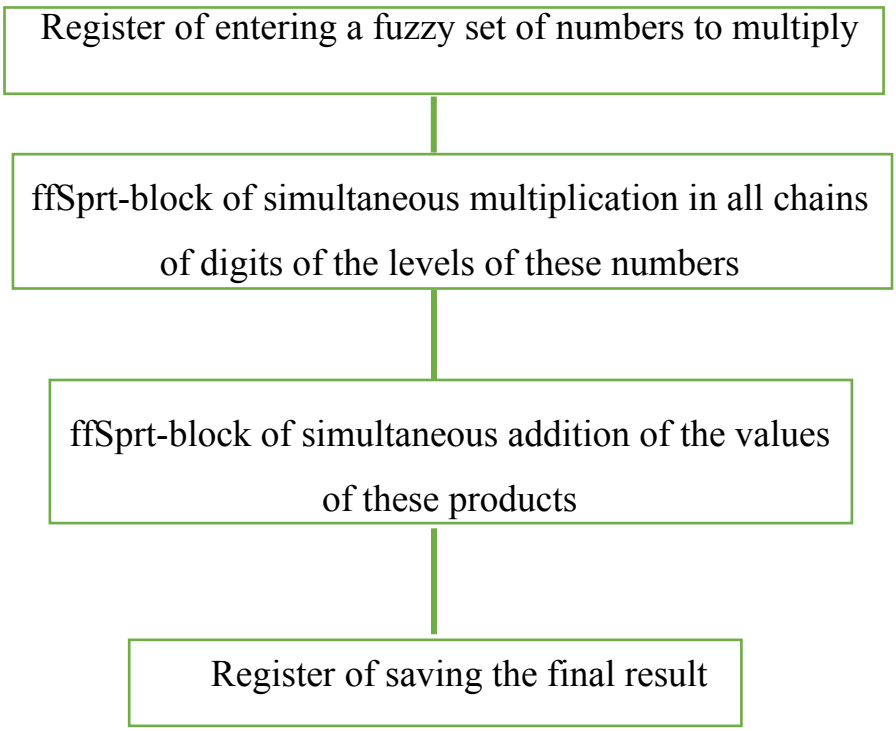


Fig. 3.2. The straightforward functional scheme of the assumed arithmetic-logical device for ffSprt-multiplication.

Remark 3.2.1. 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. 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 $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2)|_{,\dots,}x_n|\mu_{\tilde{x}}(x_n))$ and fuzzy expelling of the expressions $\tilde{r}=(r_1|\mu_{\tilde{r}}(r_1), r_2|\mu_{\tilde{r}}(r_2)|_{,\dots,}r_n|\mu_{\tilde{r}}(r_n))$ from the variables $\tilde{y}=(y_1|\mu_{\tilde{y}}(y_1), y_2|\mu_{\tilde{y}}(y_2)|_{,\dots,}y_n|\mu_{\tilde{y}}(y_n))$ simultaneously. This is implemented via

$$\begin{matrix} y_1 & y_2 & \cdots & y_n \\ \mu_{11} & \mu_{21} & \cdots & \mu_{n1} \\ := r_2 & := r_2 & \dots := & r_n \end{matrix} \text{ ffSprt } \begin{matrix} \{\{\tilde{x}\} := \} \\ \mu \\ \{p\} \end{matrix} \text{ or}$$

$$\begin{matrix} y_1 & y_2 & \cdots & y_n \\ \mu_{11} & \mu_{21} & \cdots & \mu_{n1} \\ := r_2 & := r_2 & \dots := & r_n \end{matrix} \overleftarrow{ffSprt} \begin{matrix} \{\{\tilde{x}\} := \} \\ \mu \\ \{p\} \end{matrix} \text{ for ordered case.}$$

2. Simultaneous fuzzy p-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 $\begin{matrix} \tilde{Q} \\ \mu \\ IF\{\{\tilde{B}\}\{\tilde{g}\}\} \text{ then} \end{matrix}$ ffSprt

$IF\{\{\tilde{B}\}\{\tilde{g}\}\}$ $\begin{matrix} \text{then} \\ \mu \\ \tilde{Q} \end{matrix}$ or $\begin{matrix} \tilde{Q} \\ \mu \\ IF\{\{\tilde{B}\}\{\tilde{g}\}\} \text{ then} \end{matrix}$ $\overrightarrow{ffSprt}$ $IF\{\{\tilde{B}\}\{\tilde{g}\}\}$ $\begin{matrix} \text{then} \\ \mu \\ \tilde{Q} \end{matrix}$ for ordered case. where $\tilde{Q}$ can be anything.

3. Similarly for loop operators and others.

Analogue of ffS$_3$fμ for pself– 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 analogue of form (1.1) for pself or analogues of forms (1.1.1) - (1.4) for pself, (2.1*) and analogues 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 ffSprt and analogue of ffS$_3$f(t) for pself can be used for programming:

1. The process of simultaneous fuzzy assignment and expelling 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)|$\mu_{\tilde{x}(t)}$(x$_2$(t))|$_{,\dots,}$x$_n$(t)|$\mu_{\widetilde{x(t)}}$(x$_n$(t))|). This is implemented via

$$\begin{matrix} \widetilde{w(t)} \\ \mu(t) \\ \{\{\widetilde{x(t)}\} \widetilde{:=} \{\widetilde{p(t)}\}\} \end{matrix} \text{ffSprt}(t) \begin{matrix} \{\{\widetilde{x(t)}\} \widetilde{:=} \{\widetilde{p(t)}\}\} \\ \mu(t) \\ \widetilde{w(t)} \end{matrix} \text{ or}$$

$$\begin{matrix} \widetilde{w(t)} \\ \mu(t) \\ \{\{\widetilde{x(t)}\} \widetilde{:=} \{\widetilde{p(t)}\}\} \end{matrix} \overrightarrow{\text{ffSprt}}(t) \begin{matrix} \{\{\widetilde{x(t)}\} \widetilde{:=} \{\widetilde{p(t)}\}\} \\ \mu(t) \\ \widetilde{w(t)} \end{matrix} \text{ for ordered case.}$$

2. The process of simultaneous p-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)|$\mu_{\tilde{B}(t)}$(B$_2$(t))|$_{,\dots,}$B$_n$(t)|$\mu_{\widetilde{B(t)}}$(B$_n$(t))|). Implemented via

$$\begin{matrix} \widetilde{w(t)} \\ \mu(t) \\ IF\{\{\widetilde{B(t)}\}\{\widetilde{g(t)}\}\} \ then \ \widetilde{Q(t)} \end{matrix} \text{ffSprt}(t) \begin{matrix} IF\{\{\widetilde{B(t)}\}\{\widetilde{g(t)}\}\} \ then \ \widetilde{Q(t)} \\ \mu(t) \\ \widetilde{w(t)} \end{matrix} \text{, or}$$

$$\overset{\widetilde{w}(t)}{\underset{\mu(t)}{IF\left\{\left\{B(\widetilde{t})\right\}\left\{\tilde{g}(\widetilde{t})\right\}\right.}}\ then\ Q(\widetilde{t}) \xrightarrow{\text{ffSprt}(t)} \overset{IF\left\{\left\{B(\widetilde{t})\right\}\left\{\tilde{g}(\widetilde{t})\right\}\right.}{\underset{\widetilde{w}(t)}{\mu(t)}}\ then\ Q(\widetilde{t})}$$ for

ordered case, where $Q(\widetilde{t})$ can be anything.

3. Similarly for fuzzy loop operators and others.

Analogue of $ffS_3f(t)_{\mu(t)}$ for pself– fuzzy fsoftware operators will differ only just because aggregates $x(\widetilde{t})$,$p(\widetilde{t})$, $B(\widetilde{t})$,$g(\widetilde{t})$ will be formed from corresponding processes by ffSprt-program operators in analogue of form (1.1) for pself at more complex operators in analogues of forms (1.1.1) - (1.4), (2.1*) and analogue of forms (1.1.1) - (1.4) by type (2.1*) for paself.

ffSprt (t) continual elements are used for ffSprt-coding, ffSprt-translation, fuzzy coding fpaself, and fuzzy translation fpaself for networks, which is suitable for electric current of ultrahigh frequency or using ultra-short optical pulses laser.

3.2.1 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 $\underset{x(t)q(t)}{\overset{w(t)}{\mu(t)}}$ ffSprt(t) $\underset{w(t)}{\overset{x(t)q(t)}{\mu(t)}}$ 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(designation - mnffSprt) for simple information processing. More complex executing programs are used for mnffSprt nodes. Threshold element ffSprt - $\underset{\{qy\}}{\overset{b}{\mu_2}}$ ffSprt(t) $\underset{b}{\overset{\{a\tilde{x}\}}{\mu_1}}$, b- artificial neurons of type ffSprt (designation - mnffSprt) , $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2)|,\ldots,x_n|\mu_{\tilde{x}}(x_n)|)$ is the fuzzy set of values of the initial signals, $a=(a_1,a_2,\ldots,a_n)$ are the weights of ffSprt-synapses and $\tilde{y}=(y_1|\mu_{\tilde{y}}(y_1), y_2|\mu_{\tilde{y}}(y_2)|,\ldots,y_n|\mu_{\tilde{y}}(y_n)|)$ is the fuzzy set of values of the output signals with weights $q=(q_1,q_2,\ldots,q_n)$.. The first level of mnffSprt consists of simple mnffSprt. The second level of mnffSprt consists of $\underset{\{mnSt\}}{\overset{D}{fSprt}}_{D}^{\{mnSprt\}}$ – Sprt-node of mnffSprt in range D, D- fuzzy capacity for mnffSprt node. The third level of mnffSprt consists of $\underset{ffSprt}{\overset{D}{\mu}}$ mnffSprt^{ffSprt} $\underset{D}{\overset{mnffSprt}{\mu}}$ $\underset{D}{\overset{ffSprt}{\mu}}$ - ffSprt^2- node of mnffSprt in range D, thus D becomes fuzzy capacity of pself 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 3.2.2. 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 $\overset{\text{mnffSprt}}{\underset{\{\text{ffpeprograms}\}}{\mu}}_{\text{ffSprt}}\overset{\{\text{ffpeprograms}\}}{\underset{\text{mnffSprt}}{\mu}}$, ffpeprogram –executing program in ffSprt - OS. ffSprt -OS (or fpaself-OS) is based on ffSprt -assembly language (or ffpaself-assembly language), which is based on assembly language through ffSprt-approach in turn, if the base of elements of ffSprt -networks is sufficient. The ffpeprograms are in ffSprt-programming environments (or fpaself-fuzzy programming environments), but this question and ffSprt -networks base will be considered in the following monographs. In particular, ffpeprograms may contain ffSprt- programming operators. In mnffSprt cores, the constant memory ffSprt with correspondent ffpeprograms depending on mnffSprt.

The OS (operating system) and the principles and modes of operation of the ffSprt-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 fpself-energy level, the mode $\overset{\text{ffSmnsprt}}{\underset{\text{ffSmnsprt}}{\mu}}_{\text{ffSprt}}\overset{\text{ffSmnsprt}}{\underset{\text{ffSmnsprt}}{\mu}}$ 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 ffSprt– 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 ffSprt-coding after activation. Unfortunately, triodes are not suitable for ffSprt -neural networks. In the most primitive case, usual separators with corresponding resistances and cores for ffpeprograms 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 ffpeprograms 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 ffpeprogram if one is present of needed «target weight» are taken in target block at ffSprt– activation of the networks. $\overset{activation}{\underset{ffSmnsprt,g}{\mu}}$ ffSprt $\overset{ffSmnsprt,g}{\underset{activation}{\mu}}$ or $\overset{activation}{\underset{ffSmnsprt,g}{\mu}}$ $\overrightarrow{ffSprt}\overset{ffSmnsprt,g}{\underset{activation}{\mu}}$ for ordered case derives ffSmnsprt to the ffpaself-level boundary with target weight g.

It’s used an alternating current of above high frequency and ultra-short optical pulses laser, 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."

3.3 Program operators SCprt-type and their pself-type

3.3.1 Program operators SCprt-type and their pself-type

The ideology of SCprt and SC_3f can be used for programming. Here are some of the SCprt programming operators:

1. Simultaneous assignment of the expressions $\{p\} = (p_1,p_2,...,p_n)$ to the variables $\{\mathsf{a}\} = (\mathsf{a}_1,\mathsf{a}_2,...,\mathsf{a}_n)$ and expelling expressions $\{r\} = (r_1,r_2,...,r_n)$ from variables $\{b\} = (b_1,b_2,...,b_n)$ simultaneously. This is implemented via $\overset{x}{\underset{g_1}{\{\{b\} := \{r\}\}}}\ \overset{\text{SCprt}}{}\ \overset{\{\{\mathsf{a}\} := \{p\}\}}{\underset{x}{\underset{g_1}{}}}$.
2. Simultaneous p-checking the set of conditions $\{f\} = (f_1, f_2, ...,f_n)$ for the set of expressions $\{B\} = (B_1, B_2, ...,B_n)$. Implemented via $\overset{x}{\underset{g_1}{IF\{\{B\}\{f\}\}\ then\ Q}}\ \overset{\text{SCprt}}{}\ \overset{IF\{\{B\}\{f\}\}\ then\ Q}{\underset{x}{\underset{g_1}{}}}$, where Q can be anything.
3. Similarly for loop operators and others.

p-analogue of SC_3f– software operators will differ only in that the aggregates $\{\mathsf{a}\},\{p\},\{B\},\{f\}$ 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.

The ideology of dynamic SCprt and $S_3f(t)$ can be used for programming:

1. The process of simultaneous p-assignment of the expressions $\{p(t)\} = (p_1(t),p_2(t),...,p_n(t))$ to the variables $\{g(t)\} = (g_1(t),g_2(t),...,g_n(t))$ is implemented through $\begin{matrix} x(t) \\ w(t) \\ \{\{g(t)\} := \{p(t)\}\} \end{matrix}$ $\overleftarrow{\text{.SCprt}(t)}$ $\begin{matrix} \{\{g(t)\} := \{p(t)\}\} \\ w(t) \\ x(t) \end{matrix}$.

2. The process of simultaneous p-check the set of conditions $\{f(t)\} = (f(t)_1, f_{2,}(t) ...,f(t)_n)$ for a set of expressions $\{B(t)\} = (B_1(t), B_2(t), ...,B_n(t))$ is implemented through $\begin{matrix} x(t) \\ w(t) \\ IF\{\{B(t)\}\{f(t)\}\} \ then \ Q(t) \end{matrix}$ $\overleftarrow{SCprt}$

$IF\{\{B(t)\}\{f(t)\}\} \ then \ Q(t)$

$\begin{matrix} (t) \\ \\ \end{matrix}$ $\begin{matrix} w(t) \\ x(t) \end{matrix}$ .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 $\{\{g(t)\},\{\{p(t)\}\},\{B(t)\},\{f(t)\}$ will be formed from corresponding processes 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.

3.3.2 The usage of SCprt-elements for networks

A. Galushkin's comprehensive monograph [21] 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 $\overset{a}{\underset{\{f_i x\}}{g_1}}\ SCprt\ \overset{\{f_i x\}}{\underset{a}{g_1}}$ are chosen for necessary tasks.

We will not touch on the issues of applications, or network optimization. They are described in detail by Galushkin [21]. 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 – $\mathrm{sgn}(SCprt\ \overset{\{ax\}}{\underset{b}{g_1}})$, b- mnSCprt,

$x=(x_1,x_2,\ldots,x_n)$ – source signals values, $a=(a_1,a_2,\ldots,a_n)$ – SCprt-synapses weights. The first level of mnSCprt consists of simple mnSCprt. The second level of mnSCprt consists of $\overset{D}{\underset{\{mnSCprt\}}{g_1}}\ SCprt\ \overset{\{mnSCprt\}}{\underset{D}{g_1}}$ – SCprt-node of mnSCprt in range D, D- capacity for mnSCprt node. The third level of mnSCprt consists of $SCprt\ \overset{D}{\underset{\{mnSCprt\}}{g_1}}\ SCprt\ \overset{\{mnSCprt\}}{\underset{D}{g_1}}$ - $SCprt^2$- 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 $SCprt^2$- nodes of mnSCprt, but $self_g$ -level is higher in living organisms, particularly $SCprt^n$-, $n\geq 3$. 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 $\underset{g}{\text{mnSCprt}}\ \underset{\text{SCprt}}{\{\text{ceprograms}_g\}}$, $c\text{eprogram}_g$ –executing program in SCprt-$\{\text{ceprograms}_g\}\ \underset{\text{mnSCprt}}{}$ OS. SCprt-OS (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 $\underset{\text{SmnSCprt}}{\underset{g_1}{\text{SmnSCprt}}\ \underset{\text{SCprt}}{}\ \underset{g_1}{\text{SmnSCprt}}}$ 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 Sit-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 ceprogram$_g$ if one is present of needed «target weight» are taken in target block at SCprt – activation of the networks. $\overset{activation}{g}_{SmnSCprt,f}\ SCprt\ \overset{SmnSCprt,f}{g}_{activation}$ 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, 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.”

About SfCprt and SfC_3f programming.

The ideology of SCprt and SC_3f can be used for programming. Here are some of the SfCprt programming operators:

1. Simultaneous p-assignment of the expressions $\tilde{p}=(p_1|\mu_{\tilde{p}}(p_1), p_2|\mu_{\tilde{p}}(p_2)|,\ldots,p_n|\mu_{\tilde{p}}(p_n)|)$ to the variables $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2)|,\ldots,x_n|\mu_{\tilde{x}}(x_n)|)$. to the variables $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2)|,\ldots,x_n|\mu_{\tilde{x}}(x_n)|)$. This is implemented via $\overleftarrow{SfCprt}\begin{matrix}\{\{\tilde{x}\}:=\{\tilde{p}\}\\ g_1\\ \mu_1\\ x\end{matrix}$.
2. Simultaneous p-checking the fuzzy set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2)|,\ldots,g_n|\mu_{\tilde{g}}(g_n))$ for the set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2)|,\ldots,x_n|\mu_{\tilde{B}}(B_n))$. Implemented via $\overleftarrow{SfCprt}\begin{matrix}IF\left\{\left\{\tilde{B}\right\}\{\tilde{g}\}\right\}\ then\ \tilde{Q}\\ g_1\\ \mu_1\\ x\end{matrix}$, where Q can be anything.
3. Similarly for loop operators and others.

p-analogue of $SfrC_3f-$ 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 p-analogue of form (1.1) and for more complex operators in p-analogue of form from the forms (1.1.1) - (1.4), (2.1*) and analogues 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.

The ideology of dynamic SfCprt and $SfC_3f(t)$ can be used for programming:

1. The process of simultaneous p-assignment of the expressions $\{p(t)\} = (p_1(t),p_2(t),...,p_n(t))$ to the variables $\{g(t)\} = (g_1(t),g_2(t),...,g_n(t))$ is implemented through $\begin{matrix} x(t) \\ w(t) \\ \mu(t) \\ \{\{g(t)\} := \{p(t)\}\} \end{matrix}$ $\overleftarrow{SCprt}(t)$ $\begin{matrix} \{\{g(t)\} := \{p(t)\}\} \\ w(t) \\ \mu(t) \\ x(t) \end{matrix}$.

2. The process of simultaneous p-check the set of conditions $\{f(t)\} = (f(t)_1, f_{2,}(t) ...,f(t)_n)$ for a set of expressions $\{B(t)\} = (B_1(t), B_2(t), ...,B_n(t))$ is implemented through $\begin{matrix} x(t) \\ w(t) \\ \mu(t) \\ IF\{\{B(t)\}\{f(t)\}\} \ then \ Q(t) \end{matrix}$ $\overleftarrow{SfCprt}$

$IF\{\{B(t)\}\{f(t)\}\} \ then \ Q(t)$

$\begin{matrix} (t) \\ w(t) \\ \mu(t) \\ x(t) \end{matrix}$ .where Q(t) can be any.

3. Similarly for loop operators and others.

p-analogue of $SfC_3f(t)$ – software operators will differ only in that the aggregates $\{\{g(t)\},\{\{p(t)\}\},\{B(t)\},\{f(t)\}$ will be formed from corresponding processes SfCprt(t) for the above-mentioned programming operators through p-analogue of form (1.1) or p-analogue of form from the forms (1.1.1) - (1.4), (2.1*) and analogues of forms (1.1.1) - (1.4) by type (2.1*).

3.3.3 The usage of SfCprt-elements for networks

A. Galushkin's comprehensive monograph [21] 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, $fself_g$ -coding and $fself_g$ -translation will be more stable. The target weights f_i in $\overset{a}{\underset{\{f_i x\}}{\overset{g_1}{\mu_1}}} \overleftarrow{SfCprt} \overset{\{f_i x\}}{\underset{a}{\overset{g_1}{\mu_1}}}$ are chosen for necessary tasks. We will not touch on the issues of applications, or network optimization. They are described in detail by Galushkin [21]. 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 – $\text{sgn}(\text{SCprt} \overset{\{ax\}}{\underset{b}{\overset{g_1}{\mu_1}}})$, b- mnSCprt, $x=(x_1,x_2,\ldots,x_n)$ – source signals values, $a=(a_1,a_2,\ldots,a_n)$ – SCprt-synapses weights. The first level of mnSfCprt consists of simple mnSfCprt. The second level of mnSfCprt consists of $\overset{D}{\underset{\{mnSfCprt\}}{\overset{g_1}{\mu_1}}} \text{SfCprt} \overset{\{mnSfCprt\}}{\underset{D}{\overset{g_1}{\mu_1}}}$ – SfCprt-node of mnSfCprt in range D, D- capacity for mnSfCprt node. The third level of mnSfCprt consists of $\overset{D}{\underset{\{mnSfCprt\}}{\overset{g_1}{\mu_1}}} \text{SfCprt} \overset{\{mnSfCprt\}}{\underset{D}{\overset{g_1}{\mu_1}}}$ - $SCprt^2$- 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 SfCprt^2- nodes of mnSfCprt, but fself$_g$ -level is higher in living organisms, particularly SfCprt^n-, n≥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 contains $\overset{\text{mnSCprt}}{\underset{\{\text{ceprograms}_g\}}{\overset{g}{\mu_1}}}$ SfCprt $\overset{\{\text{ceprograms}_g\}}{\underset{\text{mnSCprt}}{\overset{g}{\mu_1}}}$, $cf\text{eprogram}_g$ –executing program in SfCprt- OS. 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 cfeprograms$_g$ 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 fself_g -energy level, the mode $\overset{\mathrm{SmnSfCprt}}{\overset{g_1}{\mu_1}}\ \mathrm{SfCprt}\ \overset{\mathrm{SmnSfCprt}}{\overset{g_1}{\mu_1}}$ is used. In normal mode, it's 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 $\mathrm{cfeprograms}_g$ 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 $\mathrm{feprograms}_g$ 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 SfCprt – activation of the networks. $\overset{activation}{\overset{g}{\mu_1}}\ \text{SfCprt}\ \overset{\text{SmnSfCprt,f}}{\overset{g}{\mu_1}}$ derives SmnSfCprt to the fself$_g$ -level boundary with target weight f. It's used ultra-short optical pulses laser or an alternating current of above high frequency , 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."

3.3.4 FUZZY PROGRAM OPERATORS SfCprt**, tprS**fC**, S^1**fC**epr, S**fC**eprt$_1$**

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 SfCprt -program operator - $\overset{activation}{\overset{w}{\mu}}\ \text{SfCprt}\ \overset{Ag}{\overset{w}{\mu}}$ with type of accommodation w 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 $\overset{activation}{\underset{\mu}{g}}\ \text{SfCprt}_{\{\text{UHF AC} := Q\}}$

$\{\text{UHF AC} := Q\}$ with type of accommodation g and measure of fuzziness μ, we $\underset{\mu}{g}$ $activation$

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 fuzzy fgprogram operators:

1) SfCprt --program operators
2) **t**prSfC -program operators
3) **S[1]**fC**epr** - program operators
4) SfCeprt_1- program operators

One example is pattern recognition: SfCprt

$$\text{if SfCprt}\ \underset{\substack{\mu\\ B}}{g}\ \ \exists\ \text{SfCprt}\ \underset{\substack{\mu\\ B}}{g}\ \ image\ archive\ \ then\ Name\ of\ q\ \ \underset{\substack{\mu\\ B}}{g}\ \ q\ .$$

The example of SfCprt-program is SfCprt

$$\{\text{SfCprt}\ \underset{\substack{\mu\\ x}}{g}\ \{\{p\} := \{\mathsf{a}(x)\}\}\ ,\ \text{SfCprt}\ \underset{\substack{\mu\\ x}}{g}\ IF\{\{B\}\{f\}\}\ then\ Q\ \underset{\substack{\mu\\ x}}{g}\ Q\ ,\ \text{SfCprt}\ \underset{\substack{\mu\\ Q}}{g}\ Q\ \}\ .$$

Consider a third type of fuzzy capacity in itself - fuzzy analogue of capacity in itself. For example, based on $\overset{\tilde{q}}{\underset{\tilde{x}}{\overset{g}{\underset{\mu}{\text{SfCprt}}}}}\overset{\tilde{x}}{\underset{\tilde{q}}{\overset{g}{\underset{\mu}{}}}}$, where $\tilde{x} = (x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \ldots, x_n|\mu_{\tilde{x}}(x_n))$, i.e. n - elements at one point, we can consider p-analogue of capacity $SfC_3f\tilde{x}\mu$ (in itself with m elements from $\tilde{x}$, m<n, which is formed according to p-analogue of form from forms (1.1.1) - (1.4), (2.1*) and analogues of forms (1.1.1) - (1.4) by type (2.1*).

Consider a third type of fuzzy fgcapacity in itself. For example, based on $\overset{\tilde{q}}{\underset{\tilde{x}}{\overset{g}{\underset{\mu}{\text{SfCprt}}}}}$ $\overset{\tilde{x}}{\underset{q}{\overset{g}{\underset{\mu}{}}}}$, where $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2)|,\ldots,x_n|\mu_{\tilde{x}}(x_n)|)$ i.e. n - elements at one point, we can consider p-analogue of fuzzy fgcapacity $SfC_3f\mu$ 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 p-assignment with fuzziness m of the expressions $\tilde{p}$ $=(p_1|\mu_{\tilde{p}}(p_1), p_2|\mu_{\tilde{p}}(p_2), \ldots, p_n|\mu_{\tilde{p}}(p_n))$ to the variables $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \ldots, x_n|\mu_{\tilde{x}}(x_n))$. This is implemented via $\overset{w}{\underset{\{\{\tilde{x}\}:=\{p\}\}}{\overset{g}{\underset{\mu}{\overleftarrow{SfCprt}}}}}\overset{\{\{\tilde{x}\}:=\{p\}\}}{\underset{w}{\overset{g}{\underset{\mu}{}}}}$.
2. Simultaneous fuzzy p-checking with fuzziness m by the fuzzy set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2), \ldots, g_n|\mu_{\tilde{g}}(g_n))$ for the fuzzy set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2), \ldots, B_n|\mu_{\tilde{B}}(B_n))$ Implemented via $\overset{w}{\underset{IF\{\{\tilde{B}\}\{\tilde{g}\}\}\ then\ \tilde{Q}}{\overset{g}{\underset{\mu}{\overleftarrow{SfCprt}}}}}\overset{IF\{\{\tilde{B}\}\{\tilde{g}\}\}\ then\ \tilde{Q}}{\underset{w}{\overset{g}{\underset{\mu}{}}}}$ where $\tilde{Q}$ can be anything.
3. Similarly for fuzzy loop operators and others.

p-analogue of 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 p-analogue of form (1.1), for more complex operators in p-analogue of forms (1.1.1) - (1.4), (2.1*) and analogues of forms (1.1.1) - (1.4) by type (2.1*).

For example, $\overset{w\{R\}}{\underset{\{R\}}{\overset{g}{\underset{\mu}{}}}}$ SfCprt $\overset{\{R\}}{\underset{w\{R\}}{\overset{g}{\underset{\mu}{}}}}$ 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 ffprogram 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

$$\overset{\text{SfCprt}}{\{}\ \overset{\{\{p\}:=\{\mathsf{a}(x)\}\}}{\underset{w}{\overset{g}{\underset{\mu}{}}}}\ ,\ \overset{IF\{\{B\}\{f\}\}\ then\ Q}{\underset{\underset{w}{\overset{g}{\underset{\mu}{}}}}{\overset{g}{\underset{\mu}{}}}}\ ,\ \overset{Q}{\underset{Q}{\overset{g}{\underset{\mu}{}}}}\ \}\ .$$

The example of SfCprt-program for SmnSfCprt:

$\overset{B}{\underset{q:=}{\overset{g}{\underset{\mu}{}}}}\ \overleftarrow{SfCprt}\ \overset{q:=}{\underset{B}{\overset{g}{\underset{\mu}{}}}}$ - fuzzy p-assigning fuzzy q to fuzzy B with type of accommodation g and measure of fuzziness μ.

$\overset{B}{\underset{q:=}{\overset{g}{\underset{\mu}{}}}}\ \overleftarrow{SfCprt}\ \overset{tw:=}{\underset{d}{\overset{g}{\underset{\mu}{}}}}$ - fuzzy assigning target weight *tw* to fuzzy d with type of accommodation g and measure of fuzziness μ and fuzzy expelling of assignment of fuzzy q to fuzzy B with type of accommodation g and measure of fuzziness μ.

ffSmnsprt *activation* $\overset{\{q\}w}{\underset{\{q\}w}{\overset{g}{\underset{\mu}{}}}}$ SfCprt $\overset{\{q\}w}{\underset{f\text{fSmnsprt }activation}{\overset{g}{\underset{\mu}{}}}}$ - ffSmnsprt p-*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 fuzzy set B and fuzzy set D from s fuzzy set C, 2) fuzzy set A to a

point q, where the elements of the fuzzy sets A, B can be continuous. For example, $\overset{C}{\underset{D}{\overset{g}{\underset{\mu}{}}}}\ \text{SfCprt}\ \overset{A}{\underset{B}{\overset{g}{\underset{\mu}{}}}}$.

There are SfCprt-coding, SfCprt-translation, SfCprt-realize of fgeprograms and of the ffprograms 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 or fuzzy set A to fuzzy fuzzy set B and fuzzy set A from s fuzzy set B simultaneously etc 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, $\overset{A}{\underset{A}{\overset{g}{\underset{\mu}{}}}}\ \text{SfCprt}\ \overset{A}{\underset{A}{\overset{g}{\underset{\mu}{}}}}$ or $\overset{B}{\underset{A}{\overset{g}{\underset{\mu}{}}}}\ \text{SfCprt}\ \overset{A}{\underset{B}{\overset{g}{\underset{\mu}{}}}}$ etc.

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:

$\overset{activation}{\underset{\{\text{UHF AC} := Q\}}{\overset{g}{\underset{\mu}{}}}}\ \text{SfCprt}\ \overset{\{\text{UHF AC} := Q\}}{\underset{activation}{\overset{g}{\underset{\mu}{}}}}$. UHF AC field activation is used.

Dynamic SfCprt and $S_3fCf(t)$ programming

The ideology of dynamic SfCprt and $S_3fCf(t)$ can be used for programming:

1. Simultaneous p-assignment of the expressions $\widetilde{p(t)}=(p_1(t)|\mu_{\widetilde{p(t)}}(p_1(t)), p_2(t)|\mu_{\widetilde{p(t)}}(p_2(t)), \ldots, p_n(t)|\mu_{\widetilde{p(t)}}(p_n(t)))$ to the variables $\widetilde{x(t)}=(x_1(t)|\mu_{\widetilde{x(t)}}$

$(x_1(t))$, $x_2(t)|\mu_{\tilde{x}(t)}(x_2(t))$, ..., $x_n(t)|\mu_{\tilde{x(t)}}(x_n(t))$. This is implemented via

$$\begin{matrix} w(t) \\ g \\ \mu \\ \{\tilde{x(t)}\} := \{p(t)\} \end{matrix} \quad \text{SfCprt}(t) \quad \begin{matrix} \{\tilde{x(t)}\} := \{p(t)\} \\ g \\ \mu \\ w(t) \end{matrix} .$$

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

$$\begin{matrix} w(t) \\ g \\ \mu \\ IF\{\{\tilde{B(t)}\}\{\tilde{g}(t)\}\} \ then \ \tilde{Q(t)} \end{matrix} \quad \text{SfCprt}$$

$$IF\{\{\tilde{B(t)}\}\{\tilde{g}(t)\}\} \ then \ \tilde{Q(t)}$$

$$(t) \quad \begin{matrix} g \\ \mu \\ w(t) \end{matrix}$$

where $\tilde{Q(t)}$ can be anything.

3. Similarly for fuzzy loop operators and others.

p-analogue of $S_3fCf(t)$– fuzzy software operators will differ only just because aggregates $\tilde{x(t)}$,$\tilde{p(t)}$, $\tilde{B(t)}$,$\tilde{g(t)}$ 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 analogues of forms (1.1.1) - (1.4) by type (2.1*).

tprSfC-program operators

The ideology of tprSfC 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 $\tilde{p}=(p_1|\mu_{\tilde{p}}(p_1), p_2|\mu_{\tilde{p}}(p_2)$, ..., $p_n|\mu_{\tilde{p}}(p_n))$ 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 $\begin{matrix} \tilde{x} \\ g \\ \mu \\ =:\tilde{p} \end{matrix}$ SfCprt.

2. Simultaneous expelling check the fuzzy set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2), \ldots, g_n|\mu_{\tilde{g}}(g_n))$ for the fuzzy set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2), \ldots, B_n|\mu_{\tilde{B}}(B_n))$. It's implemented through $\overset{w}{\underset{\mu}{g}}\text{SfCprt}\ IF\{\{\tilde{B}\}\{\tilde{g}\}\}\ then\ \tilde{Q}$ 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 analogues of forms (1.1.1) - (1.4) by type (2.1*).

Consider hierarchical tprSfC-program operator

$\underset{\mu}{\overset{B}{g}}\text{SfCprt} = \left\{ \begin{matrix} D + \underset{\mu}{\overset{\{\}}{g}}\text{SfCprt} \\ A - A \cap B \\ (B - A \cap B) \end{matrix} \right\}$, 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 $\widetilde{p(t)}=(p_1(t)|\mu_{\widetilde{p(t)}}(p_1(t)), p_2(t)|\mu_{\widetilde{p(t)}}(p_2(t)), \ldots, p_n(t)|\mu_{\widetilde{p(t)}}(p_n(t)))$ from the variables $\widetilde{x(t)}=(x_1(t)|\mu_{\widetilde{x(t)}}(x_1(t)), x_2(t)|\mu_{\tilde{x}(t)}(x_2(t)), \ldots, x_n(t)|\mu_{\widetilde{x(t)}}(x_n(t)))$.is implemented through $\underset{\mu}{\overset{\tilde{x}(t)}{g}}\text{SfCprt(t)}. = :\tilde{p}(t)$

2. The process of simultaneous expelling check the fuzzy set of conditions $\widetilde{g(t)}=(g_1(t)|\mu_{\widetilde{g(t)}}(g_1(t)), g_2(t)|\mu_{\tilde{g}(t)}(g_2(t)), \ldots, g_n(t)|\mu_{\widetilde{g(t)}}(g_n(t)))$ for the fuzzy set of expressions $\widetilde{B(t)}=(B_1(t)|\mu_{\widetilde{B(t)}}(B_1(t)), B_2(t)|\mu_{\tilde{B}(t)}(B_2(t)), \ldots, B_n(t)|\mu_{\widetilde{B(t)}}(B_n(t)))$ is

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

Consider hierarchical dynamic tprSfC-program operator:

$$\overset{B(q)}{\underset{A(q)}{\overset{g}{\mu}}}\text{SfCprt(q)} = \left\{ \begin{array}{c} t(q)_{S_1fCf(A(q)\cap B(q))} + \overset{\{\}}{\underset{\mu}{g}}\text{SfCprt(q)} \\ A(q) - A(q)\cap B(q) \\ \left(B(q) - A(q)\cap B(q)\right) \end{array} \right\},$$

S^lfCepr -program operators (form $\overset{B}{\underset{D}{\overset{g_2}{\mu_2}}}\text{S}^1fCprt\overset{A}{\underset{B}{\overset{g_1}{\mu_1}}}$ - fuzzy analogue of ${}^{B}_{D}S^1t^{A}_{B}$ [11])**)**

For example, $\underset{\{=:\{p(t)\}\}}{\overset{\{\mathsf{a}(t)\}}{\overset{g_2}{\underset{\mu_2}{}}}}\text{s}^1fC\text{prt}\underset{\{\mathsf{a}(t)\}}{\overset{g_1}{\underset{\mu_1}{}}}$ $IF\left\{\left\{B(t)\right\}\left\{f(t)\right\}\right\}$ *then* $Q(t)$.

Consider hierarchical dynamic S^1fCepr-program operator: (form

$$\overset{B}{\underset{A}{\overset{g_2}{\underset{\mu_2}{}}}}\text{S}^1fCprt\overset{A}{\underset{B}{\overset{g_1}{\underset{\mu_1}{}}}}*$$

$$\overset{A}{\underset{B}{}}\quad\overset{B}{\underset{A}{}}\ ).$$

$$\overset{g_2}{\underset{\mu_2}{}}\ \text{S}^1fCprt\overset{g_1}{\underset{\mu_1}{}}$$

$$D-A \qquad B$$

SfCeprt₁- program operators (form ${}^{C}_{D}\mu_2 \mathrm{s}_1 f C p \mathrm{rt} \mu_1{}^{A}_{B} \begin{pmatrix} C & A \\ g_2 \mathrm{S}_1 f C \mathrm{rt} & g_1 \\ \mu_2 & \mu_1 \\ D & B \end{pmatrix}$ - fuzzy analogue of ${}^{C}_{D}S_1 t^{A}_{B}$ [12])**)**

${}^{a}_{a}fS_1 t^{a}_{a}$-- sample $\begin{pmatrix} self \\ oself \end{pmatrix}$-fprogram structure example.

$fSt_{t_0}\left\{ {}^{W_q fSt_{q\left({}^{a}_{a}fS_1 t^{a}_{a}\right)}}_{q\left({}^{a}_{a}fS_1 t^{a}_{a}\right)}{}^{E_q}, \; {}^{fSt_{d_r}}_{\left\{El^{d_r}\right\}} \right\}$ can be interpreted as a fprogram operator.

${}^{a}_{a}fS_1 t^{a}_{a}$ can be interpreted as a $\begin{pmatrix} self \\ oself \end{pmatrix}$-fprogram operator.

1. $\mathrm{SfCprt}\left\{ {}^{W_q fSprt_{q\left({}^{a}_{\mu_1 \mathrm{S1} f C \mathrm{rt}} {}^{a}_{g}{}_{\mu_1}\right)}}_{q\left({}^{a}_{\mu_1 \mathrm{S1} f C \mathrm{rt}} {}^{a}_{g}{}_{\mu_1}\right)}{}^{E_q}, \; {}^{fSprt_{d_r\left(E_{in} l^{d_r}\right)}}_{\{E^{ex} l^{d_r}\}} \right\}$ —fgprogram structure example, where the assemblage point d_r is the cursor, it is quite complex self—fprogram.

2. $\mathrm{SfCprt}\left\{ {}^{W_q fSprt_{q\left({}^{a}_{\mu_1 \mathrm{S1} f C \mathrm{rt}} {}^{a}_{g}{}_{\mu_1}\right)}}_{q\left({}^{a}_{\mu_1 \mathrm{S1} f C \mathrm{rt}} {}^{a}_{g}{}_{\mu_1}\right)}{}^{E_q}, \; {}^{fSprt_{d_r\left(E_{in} l^{d_r}\right)}}_{\{E^{ex} l^{d_r}\}} \right\}$ can be interpreted as a fgprogram operator.

${}^{a}_{\mu_1 \mathrm{s}_1 f C r \mathrm{t} \mu_1}{}^{a}_{a}$ can be interpreted as a $\begin{pmatrix} s_1 elf \\ os_1 elf \end{pmatrix}$-fprogram operator. $\begin{pmatrix} a & a \\ g & g \\ \mu_1 \mathrm{S1} f C \mathrm{rt} & \mu_1 \\ a & a \end{pmatrix}$-- sample $\begin{pmatrix} s_1 elf \\ os_1 elf \end{pmatrix}$-fprogram structure example.

Remark 2.8. Energy of a living organism:

$$\mathrm{Cfg}(\mathrm{r},\ \mathrm{a}(E_q)) = \mathrm{SfCprt}\left\{ {}^{q\left(\begin{smallmatrix} a & & a \\ g & & g \\ \mu_1 \mathrm{S1} fC\mathrm{rt} & & \mu_1 \\ a & & a \end{smallmatrix}\right)} W_q fSprt_q^{E_q}{}_{\left(\begin{smallmatrix} a & & a \\ g & & g \\ \mu_1 \mathrm{S1} fC\mathrm{rt} & & \mu_1 \\ a & & a \end{smallmatrix}\right)}^{g\ \mu\ t_0},\ fSprt_{d_r\left(E_{in} l^{d_r}\right)}^{\{E^{ex} l^{d_r}\}} \right\}$$

$\begin{pmatrix} a & & a \\ g & & g \\ \mu_1 \mathrm{S1} fC\mathrm{rt} & & \mu_1 \\ a & & a \end{pmatrix}$- 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,$E^{ex} l^{d_r}$ - 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.

Energy with measure of fuzziness μ_1, μ_2 of a living organism of a person:

$$\mathrm{Cfpg}(\mathrm{r},\ \mathrm{a}(E_q)) = \mathrm{SfCprt}\left\{ {}^{q\left(\begin{smallmatrix} a & & a \\ g & & g \\ \mu_1 \mathrm{S1} fC\mathrm{rt} & & \mu_1 \\ a & & a \end{smallmatrix}\right)} W_q fSprt_q^{E_q}{}_{\left(\begin{smallmatrix} a & & a \\ g & & g \\ \mu_1 \mathrm{S1} fC\mathrm{rt} & & \mu_1 \\ a & & a \end{smallmatrix}\right)}^{g\ \mu\ t_0},\ fSprt_{d_r\left(self\left(E_{in} l^{d_r}\right)\right)}^{\{E^{ex} l^{d_r}\}} \right\}$$

3.5 Introduction to FUZZY PROGRAM OPERATORS fDprt, ftprD, fD^{1}epr, fDeprt$_1$

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 Sprt-Networks [10] in one of the central departments of which a conventional computer system is located. The parallel processor is paself-type processor, fdpeprogram - fuzzy analogue of peprogram [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 - $\begin{matrix} activation \\ Q^{-1} \\ Ag \end{matrix}$ fDprt $\begin{matrix} Ag \\ action\ Q \\ activation \end{matrix}$ or $\begin{matrix} activation \\ Q^{-1} \\ Ag \end{matrix}$ $\overrightarrow{fDprt}$ $\begin{matrix} Ag \\ action\ Q \\ activation \end{matrix}$ for ordered case, where fuzzy Ag with measure of fuzziness μ_A fuzzy acts Q with measure of fuzziness μ_Q to fuzzy *activation* with measure of fuzziness $\mu_{activation}$ and performs the inverse action of Q with measure of fuzziness μ_Q from fuzzy *activation* with measure of fuzziness $\mu_{activation}$ simultaneously, 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 fdpeprogram of operator $\begin{matrix} activation \\ Q^{-1} \\ \{\text{UHF AC} := R\} \end{matrix}$ fDprt $\begin{matrix} \{\text{UHF AC} := R\} \\ action\ Q \\ activation \end{matrix}$ 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:

5) fuzzy Dprt-program operators (designation fDprt-program operators)
6) fuzzy tprD-program operators (designation ftprD-program operators)
7) fuzzy D^1epr - program operators (designation fD^1epr -program operators)
8) fuzzy $Deprt_1$- program operators (designation f$Deprt_1$-program operators)

fDprt-algorithm Example:

Simultaneous pamultiplication with measure of fuzziness μ_Q: $\underset{\tilde{x}}{\overset{\tilde{y}}{Q^{-1}}}$fDprt

multiplication $\underset{\tilde{y}}{\overset{\tilde{x}}{Q}}$, the notation of the fuzzy set B with elements

$$b_{i_1 i_2 \ldots i_n j_1 j_2 \ldots j_n} = (\underset{RR}{\overset{GG}{Q^{-1}}}\text{fDprtmultiplication } \underset{GG}{\overset{RR}{Q}})_V \text{, RR} =$$

$$(\text{ffSprt}^{\left\{x_{1_{i_1}} * \mu_{\tilde{x}}\left(x_{1_{i_1}}\right) *, x_{2_{i_2}} * \mu_{\tilde{x}}\left(x_{2_{i_2}}\right) *, \ldots, \; x_{n_{i_n}} * \mu_{\tilde{x}}\left(x_{n_{i_n}}\right)\right\}}_{\underset{q}{\mu}})_R, \text{ GG} =$$

$$(\text{ffSprt}^{\left\{y_{1_{j_1}} * \mu_{\tilde{y}}\left(y_{1_{j_1}}\right), y_{2_{j_2}} * \mu_{\tilde{y}}\left(y_{2_{j_2}}\right), \ldots, \; y_{n_{j_m}} * \mu_{\tilde{y}}\left(y_{n_{j_m}}\right)\right\}}_{\underset{h}{\mu}})_G$$

for any $\{i_1, i_2, \ldots, i_n\}$, $\{j_1, j, \ldots, j_n\}$ without repetitions, q = $\underset{w}{\overset{K}{\text{ffSprt}_{\mu}}}$, K-set of any $\{k_1 *, k_2 *, \ldots, k_n *\}$ without repeating them, k_i-any digit, i=1,2,…,n, R= $\underset{w}{\overset{\{i_1 +, i_2 +, \ldots, i_n\}}{\text{ffSprt}^{\mu}}}$, R is the index of the lower discharge, h = $\underset{w}{\overset{L}{\text{ffSprt}_{\mu}}}$, L-set of any $\{l_1 *, l_2 *, \ldots, l_m *\}$ without repeating them, l_i-any digit, i=1,2,…,m, G= $\underset{w}{\overset{\{j_1 +, j_2 +, \ldots, j_m\}}{\text{ffSprt}^{\mu}}}$, G is the index of the lower discharge, V = ffSprt

$\{i_1 + ,i_2 + ,...,i_n + j_1 + ,j_2 + ,...,j_m\}$ (we choose an index on the scale of discharges):see Table I.1.

Then $\underset{B+}{\overset{w}{\mu}}\ \text{ffSprt}\ \underset{w}{\overset{B+}{\mu}}$ gives the final result of simultaneous p-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-p-multiplication:

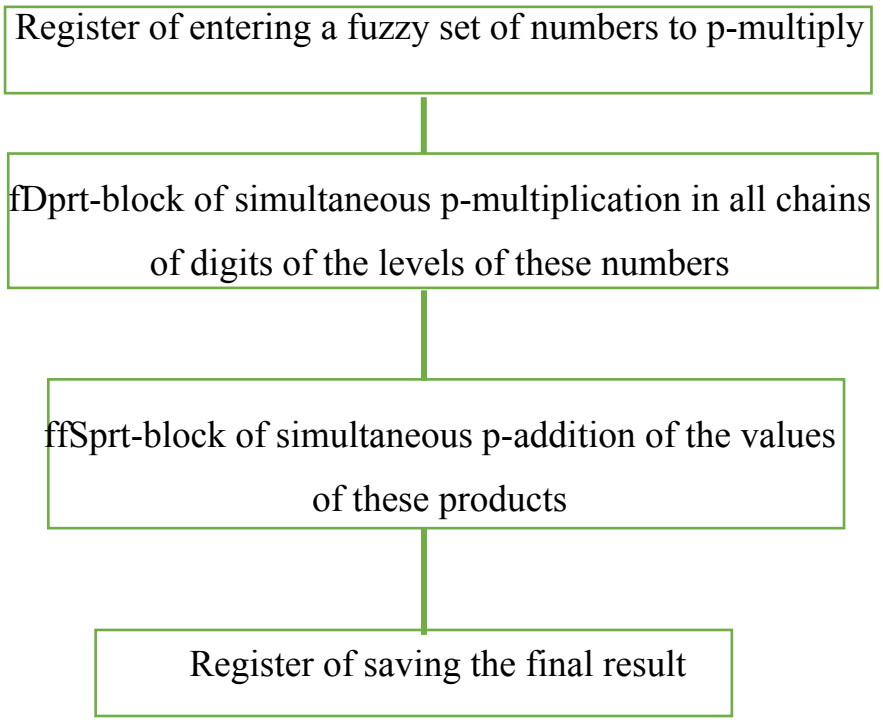


Fig. 3.5. The straightforward functional scheme of the assumed arithmetic-logical device for fDprt-p-multiplication.

Remark 3.5.1. 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 p-recognition: $\underset{\text{if ffSprt}}{\overset{\text{Name of } q}{\overset{(\text{give test result})^{-1}}{\overset{\text{image archive}}{\overset{\mu}{B}}}}} \ni \text{ffSprt}_{\overset{\mu}{B}}{}^{q}\text{fDprt}$

$\underset{\text{if ffSprt}}{\overset{\text{image archive}}{\overset{\mu}{B}}} \ni \text{ffSprt}_{\overset{\mu}{B}}{}^{q}$

give test result

$\text{Name of } q$

The example of fDprt-program is $H^{-1}{}_{\text{fDprt}}{}^{\overset{x}{RD}}$

$$\{\text{fDprt}\overset{\{\{p\}\}}{:=}\{\text{a}(x)\}, \text{fDprt}\overset{IF\{\{B\}\{f\}\}}{\underset{Q}{give\ test\ result}}, \text{fDprt}\overset{R}{action\ G}\},$$

$$\text{RD} = \{\text{fDprt}\overset{\{\{p\}\}}{:=}\{\text{a}(x)\}, \text{fDprt}\overset{IF\{\{B\}\{f\}\}}{\underset{Q}{give\ test\ result}}, \text{fDprt}\overset{R}{action\ G}\}.$$

For example, based on $Q^{-1}{}_{\text{fDprt}}{}^{\overset{\tilde{x}}{\overset{\tilde{y}}{action\ Q}}}$, where $\tilde{y}=(y_1|\mu_{\tilde{x}}(y_1), y_2|\mu_{\tilde{x}}(y_2), \ldots, y_m|\mu_{\tilde{x}}(y_m)) \subset \tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \ldots, x_n|\mu_{\tilde{x}}(x_n))$, we can consider self-type fDprt-structure – $fD_8 f\tilde{y};Q;\tilde{x}$ 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 $fD_8 f\tilde{y};Q;\tilde{x}$ 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 $\text{fDprt}_{\overset{\mu}{C}}{}^{B}$) 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 fDprt-self structure through a pseudo 3-connected form with a 2-connected form. The ideology of fDprt and $fD_8f\tilde{y};Q;\tilde{x}$ can be used for programming.

Remark 3.5.2. Fuzzy self, in particular, according to a fuzzy form is fuzzy analogue of the form of type (1.1) [1-3]: (2.1*).

Here are some of the fuzzy fDprt-program operators.

1. Simultaneous fuzzy *action* Q of the expressions $\tilde{p}=(p_1|\mu_{\tilde{p}}(p_1), p_2|\mu_{\tilde{p}}(p_2), \ldots, p_n|\mu_{\tilde{p}}(p_n))$ to the variables $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \ldots, x_n|\mu_{\tilde{x}}(x_n))$ and simultaneous fuzzy *action* Q^{-1} of the expressions $\tilde{r}=(r_1|\mu_{\tilde{r}}(r_1), r_2|\mu_{\tilde{r}}(r_2), \ldots, r_n|\mu_{\tilde{r}}(r_n))$ from the variables $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \ldots, x_n|\mu_{\tilde{x}}(x_n))$ This is implemented via $\underset{\{\tilde{r}\}}{\overset{\tilde{x}}{Q}}\text{fDprt}\underset{\{\tilde{x}\}}{\overset{\tilde{p}}{Q}}$.
2. Simultaneous R = fuzzy p-checking with fuzziness m by the fuzzy set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2), \ldots, g_n|\mu_{\tilde{g}}(g_n))$ for the fuzzy set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2), \ldots, B_n|\mu_{\tilde{B}}(B_n))$. Implemented via $\underset{\tilde{B}}{\overset{\tilde{Q}}{R^{-1}}}$ $\text{fDprt}\underset{\tilde{Q}}{\overset{\tilde{B}}{R}}$, where $\tilde{Q}$ can be anything.
3. 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 analogues of forms (1.1,1) - (1.4) by type (2.1*).

For example, $g^{-1}\underset{S}{\overset{R}{\{R,S\}}}\text{fDprt}g\underset{R}{\overset{S}{\{R,S\}}}$ is the fuzzy fDprt-pself 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 $\overleftarrow{pselff}$-fdprogram of the first type is

$$\{\ \text{fDprt}\ \underset{\{\tilde{x}\}}{\overset{\tilde{p}}{Q}},\ \text{fDprt}\ \underset{\tilde{Q}}{\overset{\tilde{B}}{R}},\ \text{fDprt}\ \underset{Q}{\overset{Q}{Q}}\ \}\ \overset{\overset{w}{R^{-1}}}{\underset{fDprt}{\longleftarrow}}\ \{\ \text{fDprt}\ \underset{\{\tilde{x}\}}{\overset{\tilde{p}}{Q}},\ \text{fDprt}\ \underset{\underset{w}{action\ R}}{\overset{\tilde{B}}{R\ \tilde{Q}}},\ \text{fDprt}\ \underset{Q}{\overset{Q}{Q}}\ \}.$$

The example of fdprt-program for fDmnsprt- fuzzy analogue of SmnSt [1-3]:

$\underset{\{\tilde{r}\}}{\overset{\tilde{x}}{Q^{-1}}}$ fDprt $\underset{\{\tilde{x}\}}{\overset{\tilde{p}}{Q}}$ - fuzzy *action* Q of $\tilde{p}$ to $\tilde{x}$.

$\underset{tw}{\overset{u}{P^{-1}}}$fDprt $\underset{g}{\overset{tw}{P}}$, where P - fuzzy assigning target weight tw to fuzzy g with measure of fuzziness μ and the reverse operation from fuzzy u simultaneously.

fDprt $\underset{f\text{Dmnsprt}\ activation}{\overset{\{q\}w}{S}}$, 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$\underset{B}{\overset{A}{Q}}$, where Q - fDprt-coding.

There are fDprt-coding, fDprt-translation, fDprt-realize of fdeprograms and fprograms from the archives without extraction theirs

fpDelf-coding.

fpDelf-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.3.1), where the elements of the fuzzy sets A, B can be continuous. For example, $\underset{A}{\overset{A}{Q^{-1}}}$fDprt$\underset{A}{\overset{A}{Q}}$.

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 fDprt-Networks, it will provide a holistic operation of the control system in three modes: conventional

serial through a conventional type of 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:

$\begin{matrix} activation \\ (:=Q)^{-1} \\ \{\text{UHF AC}\} \end{matrix}$ fDprt $\begin{matrix} \{\text{UHF AC}\} \\ :=Q \\ activation \end{matrix}$. 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 $\widetilde{p(t)}=(p_1(t)|\mu_{\widetilde{p(t)}}(p_1(t)), p_2(t)|\mu_{\widetilde{p(t)}}(p_2(t)), \ldots, p_n(t)|\mu_{\widetilde{p(t)}}(p_n(t)))$ to the variables $\widetilde{x(t)}=(x_1(t)|\mu_{\widetilde{x(t)}}(x_1(t)), x_2(t)|\mu_{\tilde{x}(t)}(x_2(t)), \ldots, x_n(t)|\mu_{\widetilde{x(t)}}(x_n(t)))$ and the reverse operation with $\{\widetilde{r(t)}\}$ from fuzzy $\widetilde{x(t)}$ simultaneously. This is implemented via $\begin{matrix} \widetilde{x(t)} \\ \widetilde{Q(t)}^{-1}\text{fDprt}(t) \\ \{\widetilde{r(t)}\} \end{matrix}$ $\begin{matrix} \widetilde{p(t)} \\ \widetilde{Q(t)} \\ \{\widetilde{x(t)}\} \end{matrix}$.
2. Simultaneous $\widetilde{R(t)}$ = fuzzy p-checking with fuzziness m by the fuzzy set of conditions $\widetilde{g(t)}=(g_1(t)|\mu_{\widetilde{g(t)}}(g_1(t)), g_2(t)|\mu_{\tilde{g}(t)}(g_2(t)), \ldots, g_n(t)|\mu_{\widetilde{g(t)}}(g_n(t)))$ for the fuzzy set of expressions $\widetilde{B(t)}=(B_1(t)|\mu_{\widetilde{B(t)}}(B_1(t)), B_2(t)|\mu_{\tilde{B}(t)}(B_2(t)), \ldots, B_n(t)|\mu_{\widetilde{B(t)}}(B_n(t)))$. Implemented via $\begin{matrix} \widetilde{Q(t)} \\ \widetilde{R(t)}^{-1}\text{f}\widetilde{Dprt}(t) \\ \widetilde{B(t)} \end{matrix}$ $\begin{matrix} \widetilde{B(t)} \\ \widetilde{R(t)} \\ \widetilde{Q(t)} \end{matrix}$, where $\tilde{Q}$ 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 analogues 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 $\tilde{p}=(p_1|\mu_{\tilde{p}}(p_1), p_2|\mu_{\tilde{p}}(p_2), \ldots, p_n|\mu_{\tilde{p}}(p_n))$ from the variables $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \ldots, x_n|\mu_{\tilde{x}}(x_n))$. This is implemented via $\underset{\{\tilde{p}\}}{\overset{\tilde{x}}{Q}}$ fDprt.
2. Simultaneous expelling R = fuzzy checking with fuzziness m by the fuzzy set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2), \ldots, g_n|\mu_{\tilde{g}}(g_n))$ for the fuzzy set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2), \ldots, B_n|\mu_{\tilde{B}}(B_n))$. It's implemented through $\underset{\tilde{B}}{\overset{\tilde{Q}}{R}}$ 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 analogues of forms (1.1,1) - (1.4) by type (2.1*).

Consider hierarchical ftprD-program operator

$(\underset{A}{\overset{B}{action\ Q}})^{-1}$Dprt $= \left\{ \begin{matrix} D + \overset{\{\}}{\underset{A-A\cap B}{\underset{(B-A\cap B)}{\mu}}} \end{matrix} \right\}$ ffSprt, where D is oself-(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 fD$_{16}$f 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 $\widetilde{p(t)}=(p_1(t)|\mu_{\widetilde{p(t)}}(p_1(t)), p_2(t)|\mu_{\widetilde{p(t)}}(p_2(t)), \ldots, p_n(t)|\mu_{\widetilde{p(t)}}(p_n(t)))$ 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 $\underset{\{\widetilde{p(t)}\}}{\overset{\widetilde{x(t)}}{Q(t)}}$ fDprt(t).

2. The process of simultaneous expelling R(t) = fuzzy checking with fuzziness m(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 $\underset{\widetilde{B(t)}}{\overset{\widetilde{Q(t)}}{R(t)}}$ fDprt(t), where $\widetilde{Q(t)}$ can be anything.

3.Similarly for loop operators and others.

$fD_{16}(t)f$ – 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 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 analogues of forms (1.1,1) - (1.4) by type (2.1*). Consider hierarchical dynamic ftprD-program operator:

$$\left(\underset{A(q)}{\overset{B(q)}{action\ Q}}\right)^{-1}\text{fDprt(q)} = \left\{ \begin{matrix} fft(q)_{S_1 f(A(q)\cap B(q))} + \overset{\{\}}{\underset{\mu}{}} \text{ffSprt(q)} \\ A(q) - A(q)\cap B(q) \\ (B(q) - A(q)\cap B(q)) \end{matrix} \right\}, \text{ where}$$

action Q- contain.

fD[1]epr -program operators (form $\left(\underset{D}{\overset{B}{action\ Q}}\right)^{-1}\text{fD}^1\text{prt}\ \underset{B}{\overset{A}{action\ Q}}$ - fuzzy analogue of ${}^{B}_{D}S^1t^{A}_{B}$ [10]))

For example, $\underset{\{\tilde{p}\}}{\overset{\tilde{x}}{R^{-1}\text{fD}^1\text{prt}}}\ \underset{\tilde{Q}}{\overset{\tilde{B}}{R}}$, where simultaneous expelling fuzzy checking with fuzziness m 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 $\tilde{p}$=($p_1|\mu_{\tilde{p}}(p_1)$, $p_2|\mu_{\tilde{p}}(p_2)$, ..., $p_n|\mu_{\tilde{p}}(p_n)$) from the variables $\tilde{x}$=($x_1|\mu_{\tilde{x}}(x_1)$, $x_2|\mu_{\tilde{x}}(x_2)$, ..., $x_n|\mu_{\tilde{x}}(x_n)$) and simultaneous R = fuzzy checking with fuzziness m by

the fuzzy set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2), \ldots, g_n|\mu_{\tilde{g}}(g_n))$ for the fuzzy set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2), \ldots, B_n|\mu_{\tilde{B}}(B_n))$, $\tilde{Q}$ can be anything.

The examples:

$(\overset{A}{action}\ \overset{A}{Q})^{-1}\text{fD}^1\text{prt}\overset{A}{action}\ \overset{A}{Q}$ can be interpreted as $\begin{pmatrix} s^1elf \\ os^1elf \end{pmatrix}$-fdprogram operator.

$(\overset{A}{action}\ \overset{A}{Q})^{-1}\text{fD}^1\text{prt}\overset{A}{action}\ \overset{A}{Q}$ sample $\begin{pmatrix} s^1elf \\ os^1elf \end{pmatrix}$-fdprogram structure example.

Consider hierarchical dynamic fD^1epr-program operator: (form

$$(\overset{B}{action}\ \overset{A}{Q})^{-1}\text{fD}^1\text{prt}\ \overset{A}{action}\ \overset{B}{Q} * \left(\overset{B}{action}\ \overset{A}{Q}\right)^{-1}\text{fD}^1\text{prt}\ \overset{A}{action}\ \overset{B}{Q}\ ).$$

fDeprt_1- program operators (form $(\overset{C}{action}\ \overset{A}{Q})^{-1}\text{fD}_1\text{prt}\overset{A}{action}\ \overset{A}{Q}$ - fuzzy analogue of ${}^{C}_{D}S_1t^{A}_{B}$ [2], [12]))

$(\overset{A}{action}\ \overset{A}{Q})^{-1}\text{fD}_1\text{prt}\overset{A}{action}\ \overset{A}{Q}$ - 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:

$$\left\{ \begin{matrix} \text{fD}_1\text{prt} \\ {}^{q}\left((\overset{A}{action}\ \overset{A}{Q})^{-1}\text{fD1prt}\overset{A}{action}\ \overset{A}{Q}\right) \\ W_q \end{matrix} fD_1prt^{E_q}_{q\left((\overset{A}{action}\ \overset{A}{Q})^{-1}\text{fD1prt}\overset{A}{action}\ \overset{A}{Q}-\right)}, \begin{matrix} E^{ex}l^{d_r} \\ \text{fD}_1\text{prt}\overset{Q}{action}\ \overset{V}{Q} \\ d_{r\left(E_{in}l^{d_r}\right)} \end{matrix} \right\}.$$

$$\mathrm{fD_1prt}\left\{ \begin{matrix} q\left({}^{A}_{A}(action\ Q)^{-1}\mathrm{fD1prt}{}^{A}_{A}action\ Q \right) \\ W_q \end{matrix} fD_1prt_q^{E_q}\left({}^{A}_{A}(action\ Q)^{-1}\mathrm{fD1prt}{}^{A}_{A}action\ Q - \right), \begin{matrix} E^{ex}l^{d_r} \\ \mathrm{fD_1prt}\,action\ Q \\ d_r\left(E_{in}l^{d_r}\right) \end{matrix} \right\}$$

$$Q$$
$$V$$

can be interpreted as a fDpr-program operator.

Appendix

Remark. Energy of a living organism:

$\mathrm{fd_1g(r,\ a(}E_q\mathrm{))} = \mathrm{fD_1prt}$

$$\left\{ \begin{matrix} q\left({}^{A}_{A}(action\ Q)^{-1}\mathrm{fD1prt}{}^{A}_{A}action\ Q \right) \\ W_q \end{matrix} fD_1prt_q^{E_q}\left({}^{A}_{A}(action\ Q)^{-1}\mathrm{fD1prt}{}^{A}_{A}action\ Q - \right), \begin{matrix} E^{ex}l^{d_r} \\ \mathrm{fD_1prt}\,action\ Q \\ d_r\left(E_{in}l^{d_r}\right) \end{matrix} \right\}$$

$$Q$$
$$V$$

$(**_{A.4})$.

Energy with measure of fuzziness μ_1, μ_2 of a living organism of a person:

$\mathrm{fd_1pg(r,\ a(}E_q\mathrm{))} = \mathrm{fD_1prt}$

$$\left\{ \begin{matrix} q\left({}^{A}_{A}(action\ Q)^{-1}\mathrm{fD1prt}{}^{A}_{A}action\ Q \right) \\ W_q \end{matrix} fD_1prt_q^{E_q}\left({}^{A}_{A}(action\ Q)^{-1}\mathrm{fD1prt}{}^{A}_{A}action\ Q - \right), \begin{matrix} E^{ex}l^{d_r} \\ \mathrm{fD_1prt}\ action\ Q \\ d_r\left(self\left(E_{in}l^{d_r}\right)\right) \end{matrix} \right\}$$

$$Q$$
$$V$$

$(***_{A.4})$.

${}^{A}_{A}(action\ Q)^{-1}\mathrm{fD_1prt}{}^{A}_{A}action\ Q$ -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. $fself^{fself^{\cdots^{fself}}}$, $\text{ff1}\downarrow \text{I} \uparrow^{1}_{-1}, ff_2{}^{\text{ff1}\downarrow \text{I}\uparrow^{1}_{-1} ff_2, \cdots^{\text{ff1}\downarrow \text{I}\uparrow^{1}_{-1} ff_2,}}$,

$fsin\infty^{fsin\infty^{\cdots^{fsin\infty}}}$.When activated in a neural network, the entire neural network becomes a working memory. Use of fself-energy as fuzzy activation or from outside. $\text{fdQ}_0=$ $\underset{\text{ffSprt}}{\overset{ffS_{mnSprt}}{\underset{activation}{\mu}}}$ $\underset{\text{fDprt}}{\overset{fD_{mnSprt}}{\underset{activation}{D}}}$ $\rightarrow$ self-ffRAM, $\text{ffQ}_{00}=$ $\underset{\text{fDprt}}{\overset{fD_{mnSprt}}{\underset{activation}{Q}}}$ fdQ_0prt, $\underset{\text{ffSprt}}{\overset{ffS_{mnSprt}}{\underset{activation}{\mu}}}$ $\underset{\text{fDprt}}{\overset{fD_{mnSprt}}{\underset{activation}{R}}}$

$\text{ffQ}_{01}=$ $\underset{\text{fDprt}}{\overset{fD_{mnSprt}}{\underset{activation}{C}}}$ $\underset{\text{fDprt}}{\overset{fD_{mnSprt}}{\underset{activation}{H}}}$ fdQ_0.fdQ_0,fdQ_{00},fdQ_{01}-coding,translation,realization in fdeprograms, use fdQ_0,fdQ_{00},fdQ_{01}-$\text{fdS}_{\text{mnSprt}}$,fdAssembler.

3.6 FUZZY PROGRAM OPERATORS Rprt, tprR, R^1epr, Reprt_1

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 Rprt -program operator - $\underset{Ag}{\overset{activation}{action\ P}}$ Rprt $\underset{activation}{\overset{Ag}{action\ Q}}$, 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 $\mu_{activation}$ and performs the inverse action of P with measure of fuzziness μ_P from fuzzy *activation* with measure of fuzziness $\mu_{activation}$ simultaneously, 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 operator $\underset{\{UHF\ AC\ :=\ R\}}{\overset{activation}{action\ P}}$ Rprt $\underset{activation}{\overset{\{UHF\ AC\ :=\ R\}}{action\ Q}}$ 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 frprogram operators:

1) fuzzy Rprt-program operators
2) fuzzy tprR-program operators
3) fuzzy R^1epr - program operators

4) fuzzy Reprt$_1$- program operators

Rprt-algorithm Example:

Simultaneous multiplication Q with measure of fuzziness μ_Q and fuzzy *action* $\tilde{P}$ with fuzzy $\tilde{U}$, $\tilde{R}$ simultaneously: $\underset{\tilde{U}}{\overset{\tilde{R}}{action}}\ \tilde{P}$ fRprt multiplication $\underset{\tilde{y}}{\overset{\tilde{x}}{Q}}$,

the notation of the fuzzy set B with elements $b_{i_1 i_2 \ldots i_n j_1 j_2 \ldots j_n} =$

$$\left(\text{ffSprt}^{\left\{x_{1_{i_1}} * \mu_{\tilde{x}}\left(x_{1_{i_1}}\right) *, x_{2_{i_2}} * \mu_{\tilde{x}}\left(x_{2_{i_2}}\right) *, \ldots,\ x_{n_{i_n}} * \mu_{\tilde{x}}\left(x_{n_{i_n}}\right)\right\}}_{\underset{q}{\mu}}\right)_R$$

$$\left(\underset{U}{\overset{R}{action}}\ P \text{ Rprt multiplication } Q\right)_V$$

$$\left(\text{ffSprt}^{\left\{y_{1_{j_1}} * \mu_{\tilde{y}}\left(y_{1_{j_1}}\right), y_{2_{j_2}} * \mu_{\tilde{y}}\left(y_{2_{j_2}}\right), \ldots,\ y_{n_{j_m}} * \mu_{\tilde{y}}\left(y_{n_{j_m}}\right)\right\}}_{\underset{h}{\mu}}\right)_G$$

for any $\{i_1, i_2, \ldots, i_n\}$, $\{j_1, j, \ldots, j_n\}$ without repetitions, $q = \underset{w}{\overset{K}{\text{ffSprt}_{\mu}}}$, K-set of any $\{k_1 *, k_2 *, \ldots, k_n *\}$ without repeating them, k_i-any digit, i=1,2,…,n, R= $\text{ffSprt}^{\{i_1 +, i_2 +, \ldots, i_n\}}_{\underset{w}{\mu}}$, R is the index of the lower discharge, $h = \underset{w}{\overset{L}{\text{ffSprt}_{\mu}}}$, L-set of any $\{l_1 *, l_2 *, \ldots, l_m *\}$ without repeating them, l_i-any digit, i=1,2,…,m, G= $\text{ffSprt}^{\{j_1 +, j_2 +, \ldots, j_m\}}_{\underset{w}{\mu}}$, G is the index of the lower discharge, V = ffSprt $\{i_1 +, i_2 +, \ldots, i_n + j_1 +, j_2 +, \ldots, j_m\}_{\underset{w}{\mu}}$ (we choose an index on the scale of discharges): see Table I.1.

Then $\text{ffSprt}^{B+}_{\underset{w}{\mu}}$ 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:

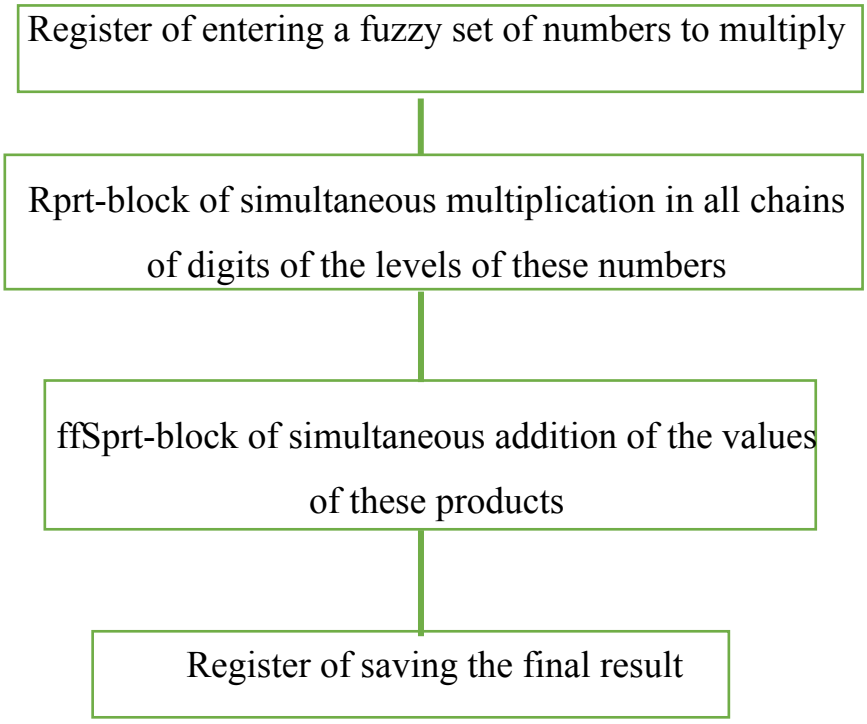


Fig. 3.6. The straightforward functional scheme of the assumed arithmetic-logical device for Rprt-multiplication.

Remark 3.6.1. 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 and *action P* with U, R simultaneously:

$$\begin{array}{c} R \\ action\ P \\ U \end{array}\ \text{Rprt}\ \begin{array}{c} \text{if ffSprt} \\ \mu \\ B \end{array}\ \begin{array}{c} image\ archive \\ \ni \text{ ffSprt} \\ give\ test\ result \\ Name\ of\ q \end{array}\ \begin{array}{c} q \\ \mu \\ B \end{array}$$

The example of Rprt-program is $\begin{array}{c} R \\ action\ P \\ U \end{array}$ Rprt

$$\{\ \text{Rprt}\ \begin{array}{c} \{\{p\}\} \\ := \\ \{a(x)\} \end{array}\ ,\ \text{Rprt}\ \begin{array}{c} IF\{\{B\}\{f\}\} \\ give\ test\ result \\ action\ H \end{array}\ ,\ \text{Rprt}\ \begin{array}{c} R \\ action\ G \\ R \end{array}\ \}\ .$$

For example, based on $\begin{array}{c} \tilde{R} \\ action\ \tilde{P} \\ \tilde{U} \end{array}$ Rprt $\begin{array}{c} \tilde{y} \\ action\ Q, \\ \tilde{x} \end{array}$ where $\tilde{y}=(y_1|\mu_{\tilde{x}}(y_1), y_2|\mu_{\tilde{x}}(y_2), \ldots, y_m|$ $\mu_{\tilde{x}}(y_m)) \subset \tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \ldots, x_n|\mu_{\tilde{x}}(x_n))$, 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 $\begin{array}{c} R \\ action\ P \\ \mu_1 \\ U \end{array}$ fRprt $\begin{array}{c} B \\ Q \\ \mu \\ C \end{array}$) 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 3.6.2. 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 $\tilde{p}=(p_1|\mu_{\tilde{p}}(p_1), p_2|\mu_{\tilde{p}}(p_2), \ldots, p_n|\mu_{\tilde{p}}(p_n))$ to the variables $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \ldots, x_n|\mu_{\tilde{x}}(x_n))$ and fuzzy *action* $\tilde{P}$ with fuzzy $\tilde{U}$, $\tilde{R}$ simultaneously. This is implemented via $\underset{\tilde{U}}{\overset{\tilde{R}}{action}}\ \tilde{P}\text{Rprt}\ \underset{\{\tilde{x}\}}{\overset{\tilde{p}}{Q}}$.
2. Simultaneous R = fuzzy checking with fuzziness m by the fuzzy set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2), \ldots, g_n|\mu_{\tilde{g}}(g_n))$ for the fuzzy set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2), \ldots, B_n|\mu_{\tilde{B}}(B_n))$ and fuzzy *action* $\tilde{P}$ with fuzzy $\tilde{U}$, $\tilde{R}$ simultaneously. Implemented via $\underset{\tilde{U}}{\overset{\tilde{R}}{action}}\ \tilde{P}\text{Rprt}\ \underset{\tilde{Q}}{\overset{\tilde{B}}{R}}$, 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 analogues of forms (1.1,1) - (1.4) by type (2.1*).

For example, $\underset{\tilde{U}}{\overset{\tilde{R}}{action}}\ \tilde{P}\text{Rprt}_{\tilde{g}}\underset{R}{\overset{S}{\{R,S\}}}$ 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 and fuzzy *action* $\tilde{P}$ with fuzzy $\tilde{U}$ simultaneously.

The example of self-frprogram of the first type is

$$\underset{\tilde{U}}{\overset{\tilde{R}}{action}}\ \tilde{P}\text{Rprt}\ \underset{\substack{action\ R\\ w}}{\overset{\tilde{Q}}{\{\text{Rprt}\ \underset{\{\tilde{x}\}}{\overset{\tilde{p}}{Q}}, \text{Rprt}\ \underset{\tilde{Q}}{\overset{\tilde{B}}{R}}, \text{Rprt}\ \underset{Q}{\overset{Q}{Q}}\}}}.$$

The example of Rprt-program operators for Rmnrprt- fuzzy analogue of SmnSt [10]:

1) $action\ \tilde{P}^{\tilde{R}}_{\tilde{U}}\text{Rprt}_{\{\tilde{x}\}}^{\tilde{p}}Q$ - fuzzy *action* Q of $\tilde{p}$ to $\tilde{x}$ and fuzzy *action* $\tilde{P}$ with fuzzy $\tilde{U}$, $\tilde{R}$ simultaneously.

2) $action\ \tilde{P}^{\tilde{R}}_{\tilde{U}}\text{Rprt}_{g}^{tw}P$, where P - fuzzy assigning target weight tw to fuzzy g with measure of fuzziness μ and fuzzy *action* $\tilde{P}$ with fuzzy $\tilde{U}$, $\tilde{R}$ simultaneously.

3) $action\ \tilde{P}^{\tilde{R}}_{\tilde{U}}\text{Rprt}_{S}^{\{q\}w}$ Rmnrprt *activation*, where S -Rmnrprt *activation* for fuzzy $\{q\}w$ with measure of fuzziness μ and fuzzy *action* $\tilde{P}$ with fuzzy $\tilde{U}$, $\tilde{R}$ simultaneously.

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, $action\ \tilde{P}^{\tilde{R}}_{\tilde{U}}\text{Rprt}_{B}^{A}Q$, 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 $\tilde{A}$, $\tilde{B}$ can be continuous. For example, $action\ \tilde{P}^{\tilde{A}}_{\tilde{A}}\text{Rprt}_{\tilde{A}}^{\tilde{A}}Q$,

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: $\overset{activation}{:= P}\ \underset{\{\text{UHF AC}\}}{\text{Rprt}}$ $\underset{\{\text{UHF AC}\}}{:= Q}$. UHF AC field activation is used.
$\overset{}{activation}$

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 $\widetilde{p(t)}=(p_1(t)|\mu_{\widetilde{p(t)}}(p_1(t)), p_2(t)|\mu_{\widetilde{p(t)}}(p_2(t)), \ldots, p_n(t)|\mu_{\widetilde{p(t)}}(p_n(t)))$ to the variables $\widetilde{x(t)}=(x_1(t)|\mu_{\widetilde{x(t)}}(x_1(t)), x_2(t)|\mu_{\tilde{x}(t)}(x_2(t)), \ldots, x_n(t)|\mu_{\widetilde{x(t)}}(x_n(t)))$ and fuzzy *action* $\widetilde{P(t)}$ with fuzzy $\widetilde{U(t)}$, $\widetilde{R(t)}$ simultaneously. This is implemented via *action* $\underset{\widetilde{U(t)}}{\overset{\widetilde{R(t)}}{\widetilde{P(t)}}}\text{Rprt}(t)\ \underset{\{\widetilde{x(t)}\}}{\overset{\widetilde{p(t)}}{\widetilde{Q(t)}}}$.
2. Simultaneous $\widetilde{R(t)}$ = fuzzy checking with fuzziness m by the fuzzy set of conditions $\widetilde{g(t)}=(g_1(t)|\mu_{\widetilde{g(t)}}(g_1(t)), g_2(t)|\mu_{\tilde{g}(t)}(g_2(t)), \ldots, g_n(t)|\mu_{\widetilde{g(t)}}(g_n(t)))$ for the fuzzy set of expressions $\widetilde{B(t)}=(B_1(t)|\mu_{\widetilde{B(t)}}(B_1(t)), B_2(t)|\mu_{\tilde{B}(t)}(B_2(t)), \ldots, B_n(t)|\mu_{\widetilde{B(t)}}(B_n(t)))$ and fuzzy *action* $\widetilde{P(t)}$ with fuzzy $\widetilde{U(t)}$, $\widetilde{R(t)}$ simultaneously. Implemented via *action* $\underset{\widetilde{U(t)}}{\overset{\widetilde{R(t)}}{\widetilde{P(t)}}}\text{Rprt}(t)\ \underset{\widetilde{Q(t)}}{\overset{\widetilde{B(t)}}{\widetilde{R(t)}}}$,

   where $\widetilde{R(t)}, \widetilde{P(t)}$ 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 analogues 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 $t_{S_4 f}$ from [14] can be used for programming. Here are some of the tprR-program operators.

1. Simultaneous expelling fuzzy *action* Q of the expressions $\tilde{p}=(p_1|\mu_{\tilde{p}}(p_1), p_2|\mu_{\tilde{p}}(p_2), \ldots, p_n|\mu_{\tilde{p}}(p_n))$ from the variables $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \ldots, x_n|\mu_{\tilde{x}}(x_n))$. This is implemented via $\underset{\{\tilde{p}\}}{\overset{\tilde{x}}{Q}}$ Rprt.
2. Simultaneous expelling R = fuzzy checking with fuzziness m by the fuzzy set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2), \ldots, g_n|\mu_{\tilde{g}}(g_n))$ for the fuzzy set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2), \ldots, B_n|\mu_{\tilde{B}}(B_n))$. It's implemented through $\underset{\tilde{B}}{\overset{\tilde{Q}}{R}}$ 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 analogues of forms (1.1,1) - (1.4) by type (2.1*).

Consider hierarchical tprR-program operator

$\underset{A}{\overset{B}{(action\ Q)^{-1}}}\text{Rprt} = \left\{ D + \underset{A - A \cap B}{\overset{\{\}}{\mu}} \text{ffSprt} \right\}$, where D 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 $\widetilde{p(t)}=(p_1(t)|\mu_{\widetilde{p(t)}}(p_1(t)), p_2(t)|\mu_{\widetilde{p(t)}}(p_2(t)), \ldots, p_n(t)|\mu_{\widetilde{p(t)}}$

$(p_n(t)))$ from the variables $\widetilde{x(t)}=(x_1(t)|\mu_{\widetilde{x(t)}}(x_1(t)), x_2(t)|\mu_{\widetilde{x(t)}}(x_2(t)), \ldots,$ $x_n(t)|\mu_{\widetilde{x(t)}}(x_n(t)))$. This is implemented via $\underset{\{\widetilde{p(t)}\}}{\overset{\widetilde{x(t)}}{Q(t)\ \text{Rprt(t)}}}$.

2. The process of simultaneous expelling R(t) = fuzzy checking with fuzziness m(t) by the fuzzy set of conditions $\widetilde{g(t)}=(g_1(t)|\mu_{\widetilde{g(t)}}(g_1(t)), g_2(t)|\mu_{\widetilde{g}(t)}(g_2(t)), \ldots, g_n(t)|\mu_{\widetilde{g(t)}}(g_n(t)))$ for the fuzzy set of expressions $\widetilde{B(t)}=(B_1(t)|\mu_{\widetilde{B(t)}}(B_1(t)), B_2(t)|\mu_{\widetilde{B}(t)}(B_2(t)), \ldots,$ $B_n(t)|\mu_{\widetilde{B(t)}}(B_n(t)))$ is implemented through $\underset{\widetilde{B(t)}}{\overset{\widetilde{Q(t)}}{R(t)\ \text{Rprt(t)}}}$, 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 analogues of forms (1.1,1) - (1.4) by type (2.1*). Consider hierarchical dynamic tprR-program operator:

$$\underset{A(q)}{\overset{B(q)}{(action\ Q)}}^{-1}\text{Rprt(q)} = \left\{ \begin{array}{c} fft(q)_{S_1 f(A(q)\cap B(q))} + \overset{\{\}}{\underset{A(q)-A(q)\cap B(q)}{\mu}}\ \text{ffSprt(q)} \\ (B(q)-A(q)\cap B(q)) \end{array} \right\},$$ where

action Q- contain.

R^1epr -program operators (form $\underset{D}{\overset{B}{(action\ Q)}}^{-1}\text{R}^1\text{prt}\ \underset{B}{\overset{A}{action\ Q}}$ - fuzzy analogue of ${}^{B}_{D}S^1t^{A}_{B}$ [10]**))**

For example, $\underset{\{\tilde{p}\}}{\overset{\tilde{x}}{R^{-1}\text{R}^1\text{prt}}}\ \underset{\tilde{Q}}{\overset{\tilde{B}}{R}}$, where simultaneous expelling fuzzy checking with fuzziness m by the fuzzy set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2), \ldots, g_n|\mu_{\tilde{g}}(g_n))$ for the fuzzy set $\tilde{p}=(p_1|\mu_{\tilde{p}}(p_1), p_2|\mu_{\tilde{p}}(p_2), \ldots, p_n|\mu_{\tilde{p}}(p_n))$ from the variables $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \ldots, x_n|\mu_{\tilde{x}}(x_n))$ and simultaneous R = fuzzy checking with fuzziness m by

the fuzzy set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2), \ldots, g_n|\mu_{\tilde{g}}(g_n))$ for the fuzzy set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2), \ldots, B_n|\mu_{\tilde{B}}(B_n))$, $\tilde{Q}$ can be anything.

The examples:

$({}^{A}_{A}action\ Q)^{-1}\mathrm{R}^1\mathrm{prt}\,{}^{A}_{A}action\ Q$ can be interpreted as $\begin{pmatrix} s^1elf \\ os^1elf \end{pmatrix}$- R^1epr-program operator. $({}^{A}_{A}action\ Q)^{-1}\mathrm{R}^1\mathrm{prt}\,{}^{A}_{A}action\ Q$ sample $\begin{pmatrix} s^1elf \\ os^1elf \end{pmatrix}$-frprogram structure example.

Consider hierarchical dynamic R^1epr-program operator: form

$$({}^{B}_{A}action\ Q)^{-1}\mathrm{R}^1\mathrm{prt}\,{}^{A}_{B}action\ Q * \left(\begin{matrix} {}^{A}_{B} \\ ({}^{B}_{D-A}action\ Q)^{-1}\mathrm{R}^1\mathrm{prt}\,{}^{A}_{B}action\ Q \end{matrix} \right).$$

Reprt_1- program operators (form $({}^{C}_{D}action\ Q)^{-1}\mathrm{R}_1\mathrm{prt}\,{}^{A}_{B}action\ Q$ - fuzzy analogue of ${}^{C}_{D}S_1t^{A}_{B}$ [12])

$({}^{A}_{A}action\ Q)^{-1}\mathrm{R}_1\mathrm{prt}\,{}^{A}_{A}action\ Q$ - 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:

$$\mathrm{fR}_1\mathrm{prt}\left\{ \begin{matrix} {}^{q}_{W_q}\left({}^{A}_{A}action\ Q\right)^{-1}\mathrm{fR}_1\mathrm{prt}\,{}^{A}_{A}action\ Q \\ fR_1prt^{E_q}_{q\left({}^{A}_{A}action\ Q\right)^{-1}\mathrm{fR}_1\mathrm{prt}\,{}^{A}_{A}action\ Q-} \\ {}^{Q}_{V} \end{matrix}, \begin{matrix} \{E^{ex}l^{d_r}\} \\ \mathrm{fR}_1\mathrm{prt}\,action\ Q \\ d_{r\left(E_{in}l^{d_r}\right)} \end{matrix} \right\}.$$

$$\text{fR}_1\text{prt}\left\{ {}^{q}_{\left({}^{A}_{A}(action\ Q)^{-1}\text{fR}_1\text{prt}{}^{A}_{A}action\ Q \right)} W_q fR_1 prt^{E_q}_{q\left({}^{A}_{A}(action\ Q)^{-1}\text{fR}_1\text{prt}{}^{A}_{A}action\ Q- \right)}, \begin{matrix} \{E^{ex}l^{d_r}\} \\ \text{fR}_1\text{prt}action\ Q \\ d_{r\left(E_{in}l^{d_r}\right)} \end{matrix} \right\}$$

$$^{Q}_{V}$$

can be interpreted as a fRpt program operator.

Supplement 3.6.1

Remark. Energy of a living organism:

$\text{fr}_1\text{g}(\text{r}, \text{a}(E_q)) = \text{fR}_1\text{prt}$

$$\left\{ {}^{q}_{\left({}^{A}_{A}(action\ Q)^{-1}\text{fR}_1\text{prt}{}^{A}_{A}action\ Q \right)} W_q fD_1 prt^{E_q}_{q\left({}^{A}_{A}(action\ Q)^{-1}\text{fR}_1\text{prt}{}^{A}_{A}action\ Q- \right)}, \begin{matrix} \{E^{ex}l^{d_r}\} \\ \text{fR}_1\text{prt}action\ Q \\ d_{r\left(E_{in}l^{d_r}\right)} \end{matrix} \right\}$$

$$^{Q}_{V}$$

$(**_{\text{B.4}})$

Energy of a living organism of a person:

$$\text{fR}_1\text{prt}\left\{ {}^{q}_{\left({}^{A}_{A}(action\ Q)^{-1}\text{fR}_1\text{prt}{}^{A}_{A}action\ Q \right)} W_q fR_1 prt^{E_q}_{q\left({}^{A}_{A}(action\ Q)^{-1}\text{fR}_1\text{prt}{}^{A}_{A}action\ Q- \right)}, \begin{matrix} \{E^{ex}l^{d_r}\} \\ \text{fR}_1\text{prt}\ action\ Q \\ d_{r(self\left(E_{in}l^{d_r}\right))} \end{matrix} \right\}$$

$$^{Q}_{V}$$

$(***_{\text{B.4}})$

$(action\ Q)^{-1}\text{fR}_1\text{prt}action\ Q$ -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. $fself^{fself^{\cdots^{fself}}}$, $\text{ff1}\downarrow \text{I} \uparrow^{1}_{-1}, ff_2^{\text{ff1}\downarrow \text{I}\uparrow^{1}_{-1} ff_2, \cdots^{\text{ff1}\downarrow \text{I}\uparrow^{1}_{-1} ff_2,}}$,

$fsin\infty^{fsin\infty^{\cdots^{fsin\infty}}}$.When activated in a neural network, the entire neural network becomes a working memory. Use of fself-energy as fuzzy activation or from outside. $\text{fdQ}_0 = \text{Rprt}^{\text{ffSprt}^{ffS_{mnSprt}}_{\mu\ activation}}_{Q} \rightarrow$ self-frRAM, $\text{frQ}_{00} = {}^{R_{mnSprt}}_{D\ activation}\text{Rprt}^{\text{ffSprt}^{ffS_{mnSprt}}_{\mu\ activation}}_{Q}\ frQ_0\text{prt}$,

$\text{frQ}_{01} = {}^{\text{Rprt}^{R_{mnSprt}}_{C\ activation}}_{\text{Rprt}^{R_{mnSprt}}_{H\ activation}} frQ_0$. frQ_0,frQ_{00},frQ_{01}-coding, translation, realization in freprograms, use frQ_0,frQ_{00},frQ_{01}-$\text{frS}_{\text{mnSprt}}$,frAssembler.

Supplement 3.4.2

Let us introduce the following notations:

$\text{A*B} = Sprt^{A}_{B}$, $\text{A}^2 = \text{Self A} = Srt^{A}_{A}$, $A^{\frac{3}{2}} = \text{Rrt}^{A}_{A} = \text{sel}f^{\frac{3}{2}}(\text{A})$, $\text{A}^3 = \text{Self}^2\text{A}$, …, $A^{\frac{3n}{2}} = \text{Rrt}^{A^n}_{A^n} = \text{sel}f^{\frac{3n}{2}}(\text{A})$, $\text{A}^{n+1} = \text{Self}^n\text{A}$, $\text{sel}f^{\min(n,m)}(\text{A}) \in Srt^{A^n}_{A^m} = \text{sel}f^{\frac{n}{m}}(\text{A})$, $\text{sel}f^{\min(n,m,k)}(\text{A}) \in \text{Rrt}^{A^n}_{A^m} = \text{sel}f^{\frac{n+m+k}{2k}}(\text{A})$, …, $\text{relf}_P(\text{A}) = {}^{A}_{A}P\text{Rprt}^{A}_{A}$, $\text{prelf} = {}^{A}_{A}A\text{Rprt}^{A}_{A}$, $\text{orelf}_P(\text{A}) = {}^{A}_{A}P\text{Rprt}$, $\text{orelf} = {}^{A}_{A}A\text{Rprt}$ etc.

There is no commutativity here: A*B ≠ B*A. We can consider operator functions: $e^A = 1 + \frac{A}{1!} + \frac{A^2}{2!} + \frac{A^3}{3!} + ...$, $(A+B)^n = \sum_{k=0}^{n} \binom{n}{k} A^k B^{n-k}$, $(1+A)^n = 1 + \frac{Ax}{1!} + \frac{n(n-1)A^2}{2!} + ...$, 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 = $PSelf^2A$, …, A^{n+1} = $PSelf^nA$, $PSelf^{\min(n,m)}(A) \in PSrt_{A^m}^{A^n}$ = $PSelf^{\frac{n}{m}}$ (A), …etc. There is no commutativity here: A*B ≠ B*A. We can consider operator functions: $e^A = 1 + \frac{A}{1!} + \frac{A^2}{2!} + \frac{A^3}{3!} + ...$, $(A+B)^n = \sum_{k=0}^{n} \binom{n}{k} A^k B^{n-k}$, $(1+A)^n = 1 + \frac{Ax}{1!} + \frac{n(n-1)A^2}{2!} + ...$, 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 $Rprt\ x = self$, 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.

3.7 Rprt-networks

A. Galushkin's comprehensive monograph [21] 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 $f = (f_1, f_2, \ldots, f_n)$ in $\underset{\{fx\}}{\overset{D}{action\ P}}\text{Rprt}\underset{D}{\overset{\{fx\}}{action\ Q}}$ are chosen for necessary tasks using a short-pulse laser to generate attosecond pulses of light to accomplish them, $x= (x_1, x_2, \ldots, x_n)$, 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 [21]. 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 mnRprt nodes. Rprt-threshold element $-\underset{\{qy\}}{\overset{b}{action\ P}}\text{Rprt}(t)\underset{b}{\overset{\{ax\}}{action\ Q}}$, Q there is a containment operation there is a containment operation, b- artificial neurons of type Rprt (designation - mnRprt), $x= (x_1, x_2, \ldots, x_n)$ are the values of the initial signals, $a=(a_1,a_2,\ldots,a_n)$ 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 $\underset{\{mnRprt\}}{\overset{D}{action\ P}}\text{Rprt}\underset{D}{\overset{\{mnRprt\}}{action\ Q}}$ – Rprt-node of mnRprt in range D, D- capacity for mnRprt node. The third level of mnRprt consists of

$$\begin{matrix} D & & \{\text{mnRprt}\} \\ \textit{action } P & & \{\text{Rprt } \textit{action } Q \} \\ \{\text{mnRprt}\} & \text{Rprt} & D \\ \{\text{Rprt } \textit{action } R \} & & \textit{action } Q \\ D & & D \end{matrix}$$ - Rprt^2- 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 Rprt^2- nodes of mnRprt, but self-level is higher in living organisms, particularly Rprt^n-, $n \geq 3$. 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 3.5.1. 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

$$\begin{matrix} \text{mnRprt} & & \{r\text{eprogram} \} \\ \textit{action } P & \text{Rprt} & \textit{action } Q \\ \{r\text{eprogram} \} & & \text{mnRprt} \end{matrix}$$, *rp*eprogram −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 $\underset{action\ P}{\overset{SmnRprt}{Rprt}}\overset{SmnRprt}{action\ Q}$, is used. In normal mode, it's planned to carry out the movement 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 block at Rprt – activation of the networks. $\begin{matrix} activation \\ action\ P \\ \text{SmnRprt,f} \end{matrix}$ Rprt $\begin{matrix} \text{SmnRprt,f} \\ action\ Q \\ activation \end{matrix}$ derives SmnRprt to the 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.

3.8 Introduction to FUZZY PROGRAM OPERATORS SDS, tSDS, fD^{1}epr, fDeprt$_{1}$.

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 fuzzy SDS -program operator - $\begin{matrix} activation \\ action\ Q^{-1} \\ Ag \\ Subject\ of\ Q \end{matrix}$ SDS $\begin{matrix} Subject\ of\ Q \\ Ag \\ action\ Q \\ activation \end{matrix}$, 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 $\mu_{activation}$ and performs the inverse action of Q with measure of fuzziness μ_Q from fuzzy *activation* with measure of fuzziness $\mu_{activation}$ simultaneously, 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 2018 year Gerard Mourou, Donna Strickland, or a combination of them. For the desired action, for example, using the direct parallel fsdprogram of operator $\begin{matrix} activation \\ action\ Q^{-1} \\ \{\text{UHF AC } := R\} \\ Subject\ of\ Q \end{matrix}$ SDS $\begin{matrix} Subject\ of\ Q \\ \{\text{UHF AC } := R\} \\ action\ Q \\ activation \end{matrix}$ 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 D^1epr - program operators (designation fD^1epr -program operators)

4) fuzzy $Deprt_1$- program operators (designation f$Deprt_1$-program operators)

SDS -algorithm Example:

Simultaneous multiplication Q with measure of fuzziness μ_Q:

SDS $\underset{\tilde{y}}{\overset{Subject\ of\ Q}{\tilde{x}}}$ multiplication Q, the notation of the fuzzy set B with elements

$b_{i_1 i_2 \ldots i_n j_1 j_2 \ldots j_n} =$

$$(\text{ffSprt}^{\left\{x_{1_{i_1}} * \mu_{\tilde{x}}\left(x_{1_{i_1}}\right) * , x_{2_{i_2}} * \mu_{\tilde{x}}\left(x_{2_{i_2}}\right) * , \ldots, \ x_{n_{i_n}} * \mu_{\tilde{x}}\left(x_{n_{i_n}}\right)\right\}}_{\underset{q}{\mu}})_R$$

$$(\text{SDS}\ \underset{\text{multiplication } Q}{Subject\ of\ Q}\ )_V$$

$$(\text{ffSprt}^{\left\{y_{1_{j_1}} * \mu_{\tilde{y}}\left(y_{1_{j_1}}\right), y_{2_{j_2}} * \mu_{\tilde{y}}\left(y_{2_{j_2}}\right), \ldots, \ y_{n_{j_m}} * \mu_{\tilde{y}}\left(y_{n_{j_m}}\right)\right\}}_{\underset{h}{\mu}})_G$$

for any $\{i_1, i_2, \ldots, i_n\}$, $\{j_1, j, \ldots, j_n\}$ without repetitions, q = $\text{ffSprt}^{K}_{\underset{w}{\mu}}$, K-set of any $\{k_1 *, k_2 *, \ldots, k_n *\}$ without repeating them, k_i-any digit, i=1,2,…,n, R= $\text{ffSprt}^{\{i_1 +, i_2 +, \ldots, i_n\}}_{\underset{w}{\mu}}$, R is the index of the lower discharge, h = $\text{ffSprt}^{L}_{\underset{w}{\mu}}$, L-set of any $\{l_1 *, l_2 *, \ldots, l_m *\}$ without repeating them, l_i-any digit, i=1,2,…,m, G= $\text{ffSprt}^{\{j_1 +, j_2 +, \ldots, j_m\}}_{\underset{w}{\mu}}$, G is the index of the lower discharge, V = ffSprt $\{i_1 +, i_2 +, \ldots, i_n + j_1 +, j_2 +, \ldots, j_m\}_{\underset{w}{\mu}}$ (we choose an index on the scale of discharges): see Table I.1.

Then $\text{ffSprt}^{B+}_{\underset{w}{\mu}}$ 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:

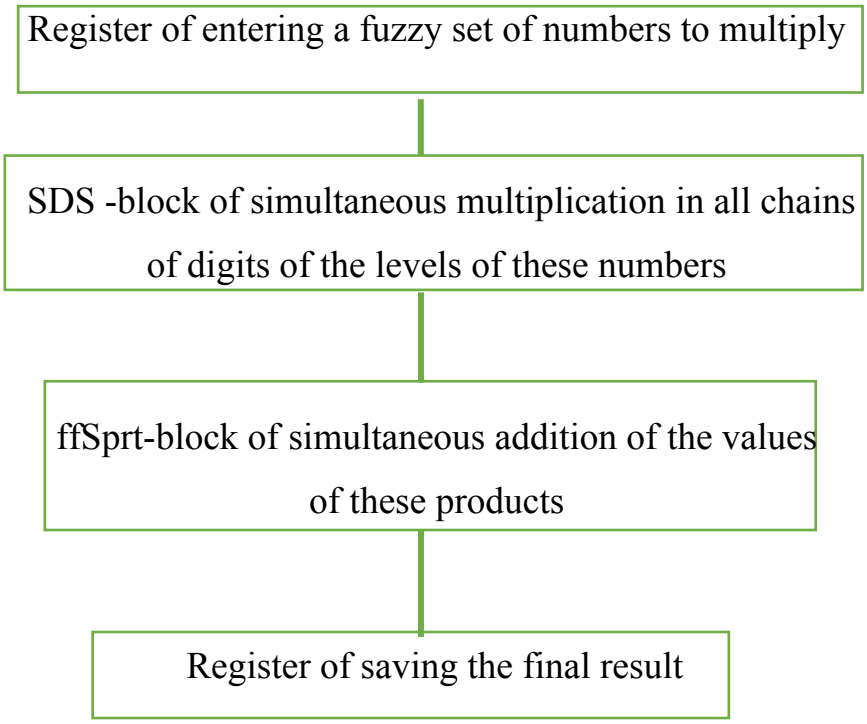


Fig. 3.8. The straightforward functional scheme of the assumed arithmetic-logical device for SDS -multiplication.

Remark 3.8.1. 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 and fuzzy *action* Q^{-1} with fuzzy $\tilde{U}$, $\tilde{R}$ simultaneously: $\begin{matrix}\tilde{R}\\ action\ Q^{-1}\\ \tilde{U}\\ Subject\ of\ Q\end{matrix}$ SDS $\begin{matrix}Subject\ of\ Q\\ image\ archive\\ \mu\\ B\\ action\ Q = give\ test\ result\\ Name\ of\ q\end{matrix}$ if ffSprt q ∋ ffSprt$_{\mu}$ B

The example of SDS -program is

$\begin{matrix}\tilde{R}\\ action\ Q^{-1}\\ \tilde{U}\\ Subject\ of\ Q\end{matrix}$ SDS { fDprt $\begin{matrix}\{\{p\}\}\\ :=\\ \{a(x)\}\end{matrix}$, fDprt $\begin{matrix}Subject\ of\ Q\\ IF\{\{B\}\{f\}\}\\ give\ test\ result\\ H\\ action\ Q\\ B\end{matrix}$, fDprt $\begin{matrix}R\\ action\ G\\ R\end{matrix}$ } .

For example, based on $\begin{matrix}\tilde{R}\\ action\ Q^{-1}\\ \tilde{U}\\ Subject\ of\ Q\end{matrix}$ SDS $\begin{matrix}Subject\ of\ Q\\ \tilde{y}\\ action\ Q\\ \tilde{x}\end{matrix}$, where $\tilde{y}$=($y_1|\mu_{\tilde{x}}(y_1)$, $y_2|\mu_{\tilde{x}}(y_2)$, ..., $y_m|\mu_{\tilde{x}}(y_m)$) ⊂ $\tilde{x}$=($x_1|\mu_{\tilde{x}}(x_1)$, $x_2|\mu_{\tilde{x}}(x_2)$, ..., $x_n|\mu_{\tilde{x}}(x_n)$), 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 $\begin{matrix}\tilde{R}\\ action\ Q^{-1}\\ \tilde{U}\\ Subject\ of\ Q\end{matrix}$ SDS $\begin{matrix}Subject\ of\ Q\\ \tilde{y}\\ action\ Q\\ \tilde{x}\end{matrix}$ 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 $\begin{matrix}\tilde{R}\\ action\ Q^{-1}\\ \tilde{U}\\ Subject\ of\ Q\end{matrix}$ SDS $\begin{matrix}Subject\ of\ Q\\ \tilde{y}\\ action\ Q\\ \tilde{x}\end{matrix}$); 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 3.8.2. 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.

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 $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \dots, x_n|\mu_{\tilde{x}}(x_n))$ and fuzzy *action* Q^{-1} with fuzzy $\tilde{U}$, $\tilde{R}$ simultaneously. This is implemented via $\begin{matrix}\tilde{R} & & Subject\ of\ Q \\ action\ Q^{-1} & \text{SDS} & \tilde{p} \\ \tilde{U} & & action\ Q \\ Subject\ of\ Q & & \tilde{x}\end{matrix}$.
2. Simultaneous R = fuzzy checking with fuzziness m 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 $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2), \dots, B_n|\mu_{\tilde{B}}(B_n))$ and fuzzy *action* R^{-1} with fuzzy $\tilde{U}$, $\tilde{W}$ simultaneously. Implemented via $\begin{matrix}\tilde{W} & & Subject\ of\ R \\ action\ R^{-1} & \text{SDS} & \tilde{B} \\ \tilde{U} & & action\ R \\ Subject\ of\ Q & & \tilde{Q}\end{matrix}$, where $\tilde{Q}$ can be anything.
3. 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 fsdprt-program operators in form (1.1) for more complex operators in forms (1.1.1) - (1.4), (2.1*) and analogues of forms (1.1.1) - (1.4) by type (2.1*).

For example, $\begin{matrix} R \\ action\ Q^{-1}\{R,\tilde{U}\} \\ \tilde{U} \\ Subject\ of\ Q\{R,\tilde{U}\} \end{matrix}$ SDS $\begin{matrix} Subject\ of\ Q\{R,S\} \\ S \\ action\ Q\{R,S\} \\ R \end{matrix}$ 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 and reverse $action\ Q^{-1}\{R,\tilde{U}\}$ from $\tilde{U}$.

The example of self-fsdprogram of the first type is

$$\begin{matrix} w \\ action\ R^{-1} \\ \tilde{U} \\ Subject\ of\ R \end{matrix} \text{ SDS } \{\ \text{fDprt}\ \begin{matrix} \tilde{p} \\ Q \\ \{\tilde{x}\} \end{matrix},\ \text{fDprt}\ \begin{matrix} \tilde{B} \\ R \\ \tilde{Q} \end{matrix},\ \text{fDprt}\ \begin{matrix} Q \\ Q \\ Q \end{matrix}\ \} \quad \begin{matrix} Subject\ of\ R \\ action\ R \\ w \end{matrix}$$

The example of fsdprogram for SDmnsd- fuzzy analogue of SmnSprt [2]:

$\begin{matrix} \tilde{W} \\ action\ Q^{-1} \\ \tilde{U} \\ Subject\ of\ Q \end{matrix}$ SDS $\begin{matrix} Subject\ of\ Q \\ \tilde{p} \\ action\ Q \\ \tilde{x} \end{matrix}$ - fuzzy *action* Q of $\tilde{p}$ to $\tilde{x}$ and fuzzy *action* R^{-1} with fuzzy $\tilde{U}$, $\tilde{W}$ simultaneously.

$\begin{matrix} \tilde{W} \\ action\ P^{-1} \\ \tilde{U} \\ Subject\ of\ Q \end{matrix}$ SDS $\begin{matrix} Subject\ of\ P \\ tw \\ action\ P \\ g \end{matrix}$, where P - fuzzy assigning target weight *tw* to fuzzy g with measure of fuzziness μ and fuzzy *action* P^{-1} with fuzzy $\tilde{U}$, $\tilde{W}$ simultaneously.

$\begin{matrix} \tilde{W} \\ action\ S^{-1} \\ \tilde{U} \\ Subject\ of\ S \end{matrix}$ SDS $\begin{matrix} Subject\ of\ S \\ \{q\}w \\ action\ S \\ \text{SDmnsd}\ activation \end{matrix}$, where S - SDmnsd *activation* for fuzzy $\{q\}w$ with measure of fuzziness μ and fuzzy *action* S^{-1} with fuzzy $\tilde{U}$, $\tilde{W}$ simultaneously.

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 example, $\begin{matrix}\tilde{W}\\ action\ Q^{-1}\\ \tilde{U}\\ Subject\ of\ Q\end{matrix}$ SDS $\begin{matrix}Subject\ of\ Q\\ A\\ action\ Q\\ B\end{matrix}$, where Q - SDS -coding.

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 sets A, B can be continuous. For example, $\begin{matrix}\tilde{W}\\ action\ Q^{-1}\\ \tilde{U}\\ Subject\ of\ Q\end{matrix}$ SDS $\begin{matrix}Subject\ of\ Q\\ A\\ action\ Q\\ A\end{matrix}$.

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 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 will be used via SDS-connectors in SDS-coding, for example:

$\begin{matrix}activation\\ action\ (:=)^{-1}\\ \{\text{UHF AC}\}\\ Subject\ of\ :=\end{matrix}$ SDS $\begin{matrix}Subject\ of\ :=\\ \{\text{UHF AC}\}\\ :=\\ activation\end{matrix}$. 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 $\widetilde{p(t)}=(p_1(t)|\mu_{\widetilde{p(t)}}(p_1(t)), p_2(t)|\mu_{\widetilde{p(t)}}(p_2(t)), \ldots, p_n(t)|\mu_{\widetilde{p(t)}}(p_n(t)))$ to the variables $\widetilde{x(t)}=(x_1(t)|$

$\mu_{x(t)}(x_1(t))$, $x_2(t)|\mu_{\tilde{x}(t)}(x_2(t))$, ..., $x_n(t)|\mu_{x(t)}(x_n(t))$) and fuzzy *action* $Q(t)^{-1}$ with fuzzy $U(t)$, $W(t)$ simultaneously. This is implemented via

$$\begin{matrix} W(t) \\ action\ Q(t)^{-1} \\ U(t) \\ Subject\ of\ Q(t) \end{matrix} \ \text{SDS} \ \begin{matrix} Subject\ of\ Q(t) \\ p(t) \\ action\ Q(t) \\ \{x(t)\} \end{matrix} .$$

2. Simultaneous $R(t)$ = fuzzy checking with fuzziness m by the fuzzy set of conditions $g(t)=(g_1(t)|\mu_{g(t)}(g_1(t)), g_2(t)|\mu_{\tilde{g}(t)}(g_2(t)), ..., g_n(t)|\mu_{g(t)}(g_n(t)))$ for the fuzzy set of expressions $B(t)=(B_1(t)|\mu_{B(t)}(B_1(t)), B_2(t)|\mu_{\tilde{B}(t)}(B_2(t)), ..., B_n(t)|\mu_{B(t)}(B_n(t)))$ and fuzzy *action* $Q(t)^{-1}$ with fuzzy $U(t)$, $W(t)$ simultaneously. Implemented via $\begin{matrix} W(t) \\ action\ Q(t)^{-1} \\ U(t) \\ Subject\ of\ Q(t) \end{matrix}$ SDS

$$\begin{matrix} Subject\ of\ R(t) \\ B(t) \\ action\ R(t) \\ Q(t) \end{matrix}$$, where $Q(t)$ can be anything.

3. Similarly for fuzzy loop operators and others.

$SD_8(t)f$ - fuzzy software operators will differ only just because aggregates $x(t)$,$p(t)$, $B(t)$,$g(t)$ 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 analogues 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.

2. Simultaneous expelling fuzzy *action Q* of the expressions $\tilde{p}=(p_1|\mu_{\tilde{p}}(p_1), p_2|\mu_{\tilde{p}}(p_2), \ldots, p_n|\mu_{\tilde{p}}(p_n))$ from the variables $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \ldots, x_n|\mu_{\tilde{x}}(x_n))$.

   This is implemented via $\begin{matrix}\tilde{x}\\ (action\ Q)^{-1}\\ \{\tilde{p}\}\\ Subject\ of\ Q\end{matrix}$ SDS.

3. Simultaneous expelling R = fuzzy checking with fuzziness m by the fuzzy set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2), \ldots, g_n|\mu_{\tilde{g}}(g_n))$ for the fuzzy set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2), \ldots, B_n|\mu_{\tilde{B}}(B_n))$. It's implemented through $\begin{matrix}\tilde{Q}\\ (action\ R)^{-1}\\ \tilde{B}\\ Subject\ of\ R\end{matrix}$ SDS, where $\tilde{Q}$ can be anything.

4. 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 analogues of forms (1.1.1) - (1.4) by type (2.1*). Consider hierarchical tSDS -program operator $\begin{matrix}B\\ (action\ Q)^{-1}\\ A\\ Subject\ of\ Q\end{matrix}$ SDS $= \left\{\begin{matrix}\{\}\\ D + \mu \text{ ffSprt}\\ A - A \cap B\\ (B - A \cap B)\end{matrix}\right\}$, 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.

4. The process of simultaneous expelling fuzzy *action Q*(*t*) of the expressions $\widetilde{p(t)}=(p_1(t)|\mu_{\widetilde{p(t)}}(p_1(t)), p_2(t)|\mu_{\widetilde{p(t)}}(p_2(t)), \ldots, p_n(t)|\mu_{\widetilde{p(t)}}$

$(p_n(t)))$ from the variables $\widetilde{x(t)}=(x_1(t)|\mu_{\widetilde{x(t)}}(x_1(t)), x_2(t)|\mu_{\widetilde{x(t)}}(x_2(t)), \ldots,$ $x_n(t)|\mu_{\widetilde{x(t)}}(x_n(t)))$. This is implemented via $\overset{\widetilde{x(t)}}{(action\ Q(t))^{-1}}\ \overset{\widetilde{p(t)}}{SDS(t)}$.

$Subject\ of\ Q(t)$

2. The process of simultaneous expelling R(t) = fuzzy checking with fuzziness m(t) by the fuzzy set of conditions $\widetilde{g(t)}=(g_1(t)|\mu_{\widetilde{g(t)}}(g_1(t)), g_2(t)|\mu_{\tilde{g}(t)}(g_2(t)), \ldots, g_n(t)|\mu_{\widetilde{g(t)}}(g_n(t)))$ for the fuzzy set of expressions $\widetilde{B(t)}=(B_1(t)|\mu_{\widetilde{B(t)}}(B_1(t)), B_2(t)|\mu_{\tilde{B}(t)}(B_2(t)), \ldots,$ $B_n(t)|\mu_{\widetilde{B(t)}}(B_n(t)))$ is implemented through $\overset{\widetilde{Q(t)}}{(action\ R(t))^{-1}}\ \overset{\widetilde{B(t)}}{SDS(t)}$, where $\widetilde{Q(t)}$ can be

$Subject\ of\ R(t)$

anything.

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 analogues of forms (1.1.1) - (1.4) by type (2.1*). Consider hierarchical dynamic tSDS-program operator:

$$\overset{B(q}{(action\ Q(q))^{-1}}\ SDS(q) = \left\{ \begin{matrix} fft(q)_{S_1 f(A(q)\cap B(q))} + \overset{\{\}}{\mu}\ ffSprt(q) \\ A(q) - A(q) \cap B(q) \\ (B(q) - A(q) \cap B(q)) \end{matrix} \right\},$$

$A(q))$

$Subject\ of\ Q(t)$

where action Q- contain.

fD[1]epr -program operators (form $\overset{B}{(action\ Q)}^{-1}\mathrm{fD^1prt}\ \overset{A}{action\ Q}$ - fuzzy analogue of $\overset{B}{D}S^1t_B^A$ [12]).

For example, $\overset{\tilde{x}}{R^{-1}}\mathrm{fD^1prt}\overset{\tilde{B}}{R}$, where simultaneous expelling fuzzy checking with $\{\tilde{p}\}\quad\tilde{Q}$

fuzziness m by the fuzzy set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2), \ldots, g_n|\mu_{\tilde{g}}(g_n))$ for

the fuzzy set $\tilde{p}=(p_1|\mu_{\tilde{p}}(p_1), p_2|\mu_{\tilde{p}}(p_2), \ldots, p_n|\mu_{\tilde{p}}(p_n))$ from the variables $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \ldots, x_n|\mu_{\tilde{x}}(x_n))$ and simultaneous R = fuzzy checking with fuzziness m by the fuzzy set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2), \ldots, g_n|\mu_{\tilde{g}}(g_n))$ for the fuzzy set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2), \ldots, B_n|\mu_{\tilde{B}}(B_n))$, $\tilde{Q}$ can be anything.

The examples:

$({}^{A}_{A}action\ Q)^{-1}\text{fD}^1\text{prt}\,{}^{A}_{A}action\ Q$ can be interpreted as $\begin{pmatrix} s^1elf \\ os^1elf \end{pmatrix}$-fdprogram operator.

$({}^{A}_{A}action\ Q)^{-1}\text{fD}^1\text{prt}\,{}^{A}_{A}action\ Q$ sample $\begin{pmatrix} s^1elf \\ os^1elf \end{pmatrix}$-fdprogram structure example.

Consider hierarchical dynamic fD^1epr-program operator: (form

$$({}^{B}_{A}action\ Q)^{-1}\text{fD}^1\text{prt}\,{}^{A}_{B}action\ Q * \left({}^{B}_{D-A}action\ Q\right)^{-1}\text{fD}^1\text{prt}\,{}^{A}_{B}action\ Q\ ).$$

fDeprt_1- program operators (form $({}^{C}_{D}action\ Q)^{-1}\text{fD}_1\text{prt}\,{}^{A}_{B}action\ Q$ - fuzzy analogue of ${}^{C}_{D}S_1t^{A}_{B}$ [12]).

$({}^{A}_{A}action\ Q)^{-1}\text{fD}_1\text{prt}\,{}^{A}_{A}action\ Q$ - sample $\begin{pmatrix} d_1self \\ d_1oself \end{pmatrix}$-fdprogram structure example.

Appendix

Entire neural network as instantaneous simultaneous SDS-RAM in SDS -elements and fself- elements. $fself^{fself^{\cdots^{fself}}}$, $\text{ff1}\downarrow \text{I} \uparrow^1_{-1}, ff_2{}^{\text{ff1}\downarrow \text{I}\uparrow^1_{-1}ff_2,\ \cdots^{\text{ff1}\downarrow \text{I}\uparrow^1_{-1}ff_2,}}$, $fsin\infty^{fsin\infty^{\cdots^{fsin\infty}}}$.When activated in a neural network, the entire neural network becomes a working memory. Use of fself-energy as fuzzy activation or from outside.

3.9 Introduction to FUZZY PROGRAM OPERATORS SRS, tSRS, fR[1]epr, fReprt[1]

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 fuzzy SRS -program operator - $\begin{matrix} activation \\ action\ P \\ Ag \\ Subject\ of\ P \end{matrix}$ SRS $\begin{matrix} Subject\ of\ Q \\ Ag \\ action\ Q \\ activation \end{matrix}$, 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 $\mu_{activation}$ and performs the inverse action of P with measure of fuzziness μ_P from fuzzy *activation* with measure of fuzziness $\mu_{activation}$ simultaneously, Q, P are any fuzzy *action*s, 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 operator $\begin{matrix} activation \\ action\ P \\ \{\text{UHF AC} := R\} \\ Subject\ of\ P \end{matrix}$ SRS $\begin{matrix} Subject\ of\ Q \\ \{\text{UHF AC} := R\} \\ action\ Q \\ activation \end{matrix}$ 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 fsrprogram operators:

5) fuzzy SRS-program operators
6) fuzzy tSRS -program operators
7) fuzzy R[1]epr - program operators (designation fR[1]epr -program operators)
8) fuzzy Reprt$_1$- program operators (designation fReprt$_1$-program operators)

SRS -algorithm Example:

Simultaneous multiplication Q with measure of fuzziness μ_Q and fuzzy *action P* with fuzzy $\tilde{U}$, $\tilde{R}$ simultaneously:

$\begin{matrix}\tilde{R} \\ action\ P \\ \tilde{U} \\ Subject\ of\ P\end{matrix}$ SRS $\begin{matrix}Subject\ of\ Q \\ \tilde{x} \\ \text{multiplication } Q \\ \tilde{y}\end{matrix}$, the notation of the fuzzy set B with elements $b_{i_1 i_2 \ldots i_n j_1 j_2 \ldots j_n} =$

$$\left(\begin{matrix}\tilde{R} \\ action\ P \\ \tilde{U} \\ Subject\ of\ P\end{matrix} \text{ SRS } \begin{matrix}Subject\ of\ Q \\ (\text{ffSprt}^{\left\{x_{1_{i_1}} * \mu_{\tilde{x}}\left(x_{1_{i_1}}\right) *, x_{2_{i_2}} * \mu_{\tilde{x}}\left(x_{2_{i_2}}\right) *, \ldots,\ x_{n_{i_n}} * \mu_{\tilde{x}}\left(x_{n_{i_n}}\right)\right\}}_{\mu\ q})_R \\ \text{multiplication } Q \\ (\text{ffSprt}^{\left\{y_{1_{j_1}} * \mu_{\tilde{y}}\left(y_{1_{j_1}}\right), y_{2_{j_2}} * \mu_{\tilde{y}}\left(y_{2_{j_2}}\right), \ldots,\ y_{n_{j_m}} * \mu_{\tilde{y}}\left(y_{n_{j_m}}\right)\right\}}_{\mu\ h})_G\end{matrix}\right)_V$$

for any $\{i_1, i_2, \ldots, i_n\}$, $\{j_1, j, \ldots, j_n\}$ without repetitions, q = $\text{ffSprt}^{K}_{\mu\ w}$, K-set of any $\{k_1 *, k_2 *, \ldots, k_n *\}$ without repeating them, k_i-any digit, i=1,2,…,n, R= $\text{ffSprt}^{\{i_1 +, i_2 +, \ldots, i_n\}}_{\mu\ w}$, R is the index of the lower discharge, h = $\text{ffSprt}^{L}_{\mu\ w}$, L-set of any $\{l_1 *, l_2 *, \ldots, l_m *\}$ without repeating them, l_i-any digit, i=1,2,…,m, G= $\text{ffSprt}^{\{j_1 +, j_2 +, \ldots, j_m\}}_{\mu\ w}$, G is the index of the lower discharge, V = ffSprt

$\underset{w}{\overset{\mu}{\{i_1 + ,i_2 + ,...,i_n + j_1 + ,j_2 + ,...,j_m\}}}$ (we choose an index on the scale of discharges): see Table I.1.

Then $\underset{w}{\overset{\mu}{\text{ffSprt}}}^{B+}$ 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:

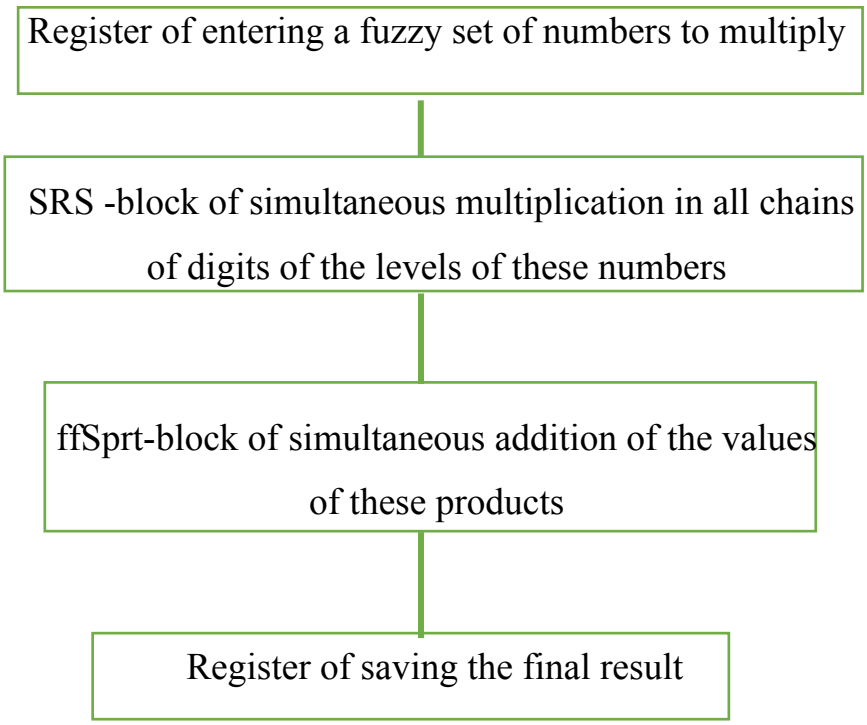


Fig. 3.9. The straightforward functional scheme of the assumed arithmetic-logical device for SRS -multiplication.

Remark.3.9.1. 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 and fuzzy *action P* with fuzzy $\tilde{U}$, $\tilde{R}$ simultaneously:

$$\begin{array}{c} \tilde{R} \\ action\ P \\ \tilde{U} \\ Subject\ of\ P \end{array} \text{ SRS } \begin{array}{c} Subject\ of\ Q \\ image\ archive \\ \text{if ffSprt}_{\mu_B} \\ action\ Q = give\ test\ result \\ Name\ of\ q \end{array} \begin{array}{c} q \\ \text{э ffSprt}_{\mu_B} \end{array}$$

The example of SRS -program is

$$\begin{array}{c} \tilde{R} \\ action\ P \\ \tilde{U} \\ Subject\ of\ P \end{array} \text{ SRS } \{ \text{fRprt} \begin{array}{c} \{\{p\}\} \\ := \\ \{a(x)\} \end{array} , \text{fRprt} \begin{array}{c} Subject\ of\ Q \\ IF\{\{B\}\{f\}\} \\ give\ test\ result \\ H \\ action\ Q \\ B \end{array} , \text{fRprt} \begin{array}{c} R \\ action\ G \\ R \end{array} \} .$$

For example, based on $\begin{array}{c} \tilde{R} \\ action\ P \\ \tilde{U} \\ Subject\ of\ P \end{array}$ SRS $\begin{array}{c} Subject\ of\ Q \\ \tilde{y} \\ action\ Q \\ \tilde{x} \end{array}$, where $\tilde{y}=(y_1|\mu_{\tilde{x}}(y_1), y_2|\mu_{\tilde{x}}(y_2), \ldots, y_m|\mu_{\tilde{x}}(y_m)) \subset \tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \ldots, x_n|\mu_{\tilde{x}}(x_n))$, 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

$\begin{array}{c} \tilde{R} \\ action\ P \\ \tilde{U} \\ Subject\ of\ P \end{array}$ SRS $\begin{array}{c} Subject\ of\ Q \\ \tilde{y} \\ action\ Q \\ \tilde{x} \end{array}$ 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 $\begin{matrix}\tilde{R}\\ action\ P\\ \tilde{U}\\ Subject\ of\ P\end{matrix}$ SRS $\begin{matrix}Subject\ of\ Q\\ \tilde{y}\\ action\ Q\\ \tilde{x}\end{matrix}$); 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 3.9.1. 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.

1. Simultaneous fuzzy *action Q* of the expressions $\tilde{p}=(p_1|\mu_{\tilde{p}}(p_1), p_2|\mu_{\tilde{p}}(p_2), ..., p_n|\mu_{\tilde{p}}(p_n))$ to the variables $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), ..., x_n|\mu_{\tilde{x}}(x_n))$ and fuzzy *action P* with fuzzy $\tilde{U}$, $\tilde{R}$ simultaneously. This is implemented via $\begin{matrix}\tilde{R}\\ action\ P\\ \tilde{U}\\ Subject\ of\ P\end{matrix}$ SRS $\begin{matrix}Subject\ of\ Q\\ \tilde{p}\\ action\ Q\\ \tilde{x}\end{matrix}$.
2. Simultaneous R = fuzzy checking with fuzziness m 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 $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2), ..., B_n|\mu_{\tilde{B}}(B_n))$ and fuzzy *action P* with fuzzy $\tilde{U}$, $\tilde{R}$ simultaneously. Implemented via $\begin{matrix}\tilde{R}\\ action\ P\\ \tilde{U}\\ Subject\ of\ P\end{matrix}$ SRS $\begin{matrix}Subject\ of\ R\\ \tilde{B}\\ action\ R\\ \tilde{Q}\end{matrix}$, where $\tilde{Q}$ can be anything.
3. 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 analogues of forms (1.1.1) - (1.4) by type (2.1*).

For example, $\begin{matrix}\tilde{R}\\ action\ P\\ \tilde{U}\\ Subject\ of\ P\end{matrix}$ SRS $\begin{matrix}Subject\ of\ Q\{R,S\}\\ S\\ action\ Q\{R,S\}\\ R\end{matrix}$ 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

$$\begin{matrix}\tilde{R}\\ action\ P\\ \tilde{U}\\ Subject\ of\ P\end{matrix} \text{ SRS } \{\ \text{fRprt}_{\begin{matrix}\tilde{p}\\ Q\\ \{\tilde{x}\}\end{matrix}}\ ,\ \text{fRprt}_{\begin{matrix}Subject\ of\ R\\ \tilde{B}\\ R\\ \tilde{Q}\\ action\ R\\ w\end{matrix}}\ ,\ \text{fRprt}_{\begin{matrix}Q\\ Q\end{matrix}}\ \}$$

The example of fsrprogram for SRmnsr- fuzzy analogue of SmnSt [10]:

$\begin{matrix}\tilde{R}\\ action\ P\\ \tilde{U}\\ Subject\ of\ P\end{matrix}$ SRS $\begin{matrix}Subject\ of\ Q\\ \tilde{p}\\ action\ Q\\ \tilde{x}\end{matrix}$ - fuzzy *action Q* of $\tilde{p}$ to $\tilde{x}$ and fuzzy *action P* with fuzzy $\tilde{U}$, $\tilde{R}$ simultaneously.

$\begin{matrix}\tilde{V}\\ action\ P\\ \tilde{U}\\ Subject\ of\ P\end{matrix}$ SRS $\begin{matrix}Subject\ of\ R\\ tw\\ action\ R\\ g\end{matrix}$, where R - fuzzy assigning target weight *tw* to fuzzy g with measure of fuzziness μ and fuzzy *action P* with fuzzy $\tilde{V}$, $\tilde{U}$ simultaneously.

$\begin{matrix}\tilde{R}\\ action\ P\\ \tilde{U}\\ Subject\ of\ P\end{matrix}$ SRS $\begin{matrix}Subject\ of\ S\\ \{q\}w\\ action\ S\\ \text{SRmnsr } activation\end{matrix}$, where S - SRmnsr *activation* for fuzzy $\{q\}w$ with measure of fuzziness μ and fuzzy *action P* with fuzzy $\tilde{U}$, $\tilde{R}$ simultaneously.

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 example, $\begin{matrix}\tilde{V}\\ action\ P\\ \tilde{U}\\ Subject\ of\ P\end{matrix}$ SRS $\begin{matrix}Subject\ of\ Q\\ A\\ action\ Q\\ B\end{matrix}$, where Q, P - 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 sets A, B can be continuous. For example, $\begin{matrix}\tilde{V}\\ action\ P\\ \tilde{U}\\ Subject\ of\ P\end{matrix}$ SRS $\begin{matrix}Subject\ of\ Q\\ A\\ action\ Q\\ A\end{matrix}$.

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:

$\begin{matrix}\tilde{V}\\ action\ P\\ \tilde{U}\\ Subject\ of\ P\end{matrix}$ SRS $\begin{matrix}Subject\ of\ :=\\ \{\text{UHF AC}\}\\ :=\\ activation\end{matrix}$. 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 $\widetilde{p(t)}=(p_1(t)|\mu_{\widetilde{p(t)}}(p_1(t)), p_2(t)|\mu_{\widetilde{p(t)}}(p_2(t)), \ldots, p_n(t)|\mu_{\widetilde{p(t)}}(p_n(t)))$ to the variables $\widetilde{x(t)}=(x_1(t)|\mu_{\widetilde{x(t)}}(x_1(t)), x_2(t)|\mu_{\widetilde{x(t)}}(x_2(t)), \ldots, x_n(t)|\mu_{\widetilde{x(t)}}(x_n(t)))$ and fuzzy *action P* with fuzzy $\tilde{V}$, $\tilde{U}$ simultaneously. This is implemented via

$$\begin{matrix}\tilde{V}\\ action\ P\\ \tilde{U}\\ Subject\ of\ P\end{matrix}\ \text{SRS}\ \begin{matrix}Subject\ of\ \widetilde{Q(t)}\\ \widetilde{p(t)}\\ action\ \widetilde{Q(t)}\\ \{\widetilde{x(t)}\}\end{matrix}\ .$$

2. Simultaneous $\widetilde{R(t)}$ = fuzzy checking with fuzziness m by the fuzzy set of conditions $\widetilde{g(t)}=(g_1(t)|\mu_{\widetilde{g(t)}}(g_1(t)), g_2(t)|\mu_{\widetilde{g(t)}}(g_2(t)), \ldots, g_n(t)|\mu_{\widetilde{g(t)}}(g_n(t)))$ for the fuzzy set of expressions $\widetilde{B(t)}=(B_1(t)|\mu_{\widetilde{B(t)}}(B_1(t)), B_2(t)|\mu_{\widetilde{B(t)}}(B_2(t)), \ldots, B_n(t)|\mu_{\widetilde{B(t)}}(B_n(t)))$ and fuzzy *action P* with fuzzy $\tilde{V}$, $\tilde{U}$ simultaneously. Implemented via $\begin{matrix}\tilde{V}\\ action\ P\\ \tilde{U}\\ Subject\ of\ P\end{matrix}$ SRS $\begin{matrix}Subject\ of\ \widetilde{R(t)}\\ \widetilde{B(t)}\\ action\ \widetilde{R(t)}\\ \widetilde{Q(t)}\end{matrix}$, where $\widetilde{Q(t)}$ 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 analogues 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 $\tilde{p}$=($p_1|\mu_{\tilde{p}}(p_1)$, $p_2|\mu_{\tilde{p}}(p_2)$, ..., $p_n|\mu_{\tilde{p}}(p_n)$) 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 $\begin{matrix}\tilde{x}\\(action\ Q)^{-1}\\\{\tilde{p}\}\\Subject\ of\ Q\end{matrix}$ SRS.
2. Simultaneous expelling R = fuzzy checking with fuzziness m 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 $\begin{matrix}\tilde{Q}\\(action\ R)^{-1}\\\tilde{B}\\Subject\ of\ R\end{matrix}$ SRS, where $\tilde{Q}$ 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 analogues of forms (1.1.1) - (1.4) by type (2.1*).

Consider hierarchical tSRS -program operator

$\begin{matrix}B\\(action\ Q)^{-1}\\A\\Subject\ of\ Q\end{matrix}$ SRS = $\left\{\begin{matrix}\{\}\\D+\quad\mu\quad ffSprt\\A-A\cap B\\(B-A\cap B)\end{matrix}\right\}$, where D 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.

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 $\widetilde{x(t)}$=($x_1(t)|\mu_{\widetilde{x(t)}}(x_1(t))$, $x_2(t)|\mu_{\tilde{x}(t)}(x_2(t))$, ..., $x_n(t)|\mu_{\widetilde{x(t)}}$ ($x_n(t)$)). This is implemented via $\overset{\widetilde{x(t)}}{(action\ Q(t))^{-1}}$ SRS(t).
$\widetilde{p(t)}$
$Subject\ of\ Q(t)$

2. The process of simultaneous expelling R(t) = fuzzy checking with fuzziness m(t) by the fuzzy set of conditions $\widetilde{g(t)}$=($g_1(t)|\mu_{\widetilde{g(t)}}(g_1(t))$, $g_2(t)|\mu_{\tilde{g}(t)}(g_2(t))$, ..., $g_n(t)|\mu_{\widetilde{g(t)}}$ ($g_n(t)$)) for the fuzzy set of expressions $\widetilde{B(t)}$=($B_1(t)|\mu_{\widetilde{B(t)}}(B_1(t))$, $B_2(t)|\mu_{\tilde{B}(t)}(B_2(t))$, ..., $B_n(t)|\mu_{\widetilde{B(t)}}(B_n(t))$) is implemented through $\overset{\widetilde{Q(t)}}{(action\ R(t))^{-1}}$ SRS(t), where $\widetilde{Q(t)}$ can be
$\widetilde{B(t)}$
$Subject\ of\ R(t)$

anything.

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 analogues of forms (1.1.1) - (1.4) by type (2.1*). Consider hierarchical dynamic tSRS-program operator:

$\overset{B(q}{(action\ Q(q))^{-1}}$ SRS(q) $= \left\{ \begin{matrix} fft(q)_{S_1 f(A(q)\cap B(q))} + \overset{\{\}}{\mu} \text{ ffSprt(q)} \\ A(q) - A(q) \cap B(q) \\ (B(q) - A(q) \cap B(q)) \end{matrix} \right\}$,
$A(q))$
$Subject\ of\ Q(t)$

where action Q- contain.

fR¹epr -program operators (form $\overset{B}{(action\ Q)}^{-1}$fR¹prt $\overset{A}{action\ Q}$ - fuzzy analogue of
D B

$\overset{B}{D}S^1t^A_B$ [10]).

For example, $\overset{\tilde{x}}{R^{-1}}$fR¹prt $\overset{\tilde{B}}{R}$, where simultaneous expelling fuzzy checking with
$\{\tilde{p}\}$ $\tilde{Q}$

fuzziness m 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 $\tilde{p}$=($p_1|\mu_{\tilde{p}}(p_1)$, $p_2|\mu_{\tilde{p}}(p_2)$, ..., $p_n|\mu_{\tilde{p}}(p_n)$) from the variables $\tilde{x}$=($x_1|\mu_{\tilde{x}}(x_1)$, $x_2|\mu_{\tilde{x}}(x_2)$, ..., $x_n|\mu_{\tilde{x}}(x_n)$) and simultaneous R = fuzzy checking with fuzziness m 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 $\tilde{B}$=($B_1|\mu_{\tilde{B}}(B_1)$, $B_2|\mu_{\tilde{B}}(B_2)$, ..., $B_n|\mu_{\tilde{B}}(B_n)$), $\tilde{Q}$ can be anything.

Appendix

Entire neural network as instantaneous simultaneous SRS-RAM in SRS -elements and fself- elements. $fself^{fself^{\cdots^{fself}}}$, $\mathrm{ff1}\downarrow \mathrm{I} \uparrow^1_{-1}, ff_2^{\mathrm{ff1}\downarrow \mathrm{I}\uparrow^1_{-1} ff_2, \cdots^{\mathrm{ff1}\downarrow \mathrm{I}\uparrow^1_{-1} ff_2,}}$, $fsin\infty^{fsin\infty^{\cdots^{fsin\infty}}}$. When activated in a neural network, the entire neural network becomes a working memory. Use of fself-energy as fuzzy activation or from outside.

3.10 Program operators PrSprt, PrtSpr, S^1e, Set_1 and their pself-type

Similarly, for simultaneous parallel multiplication and *solution of task Q_j by D_j* simultaneously: $\begin{matrix} task\ Q_1 & task\ Q_i & \dots & task\ Q_n \\ h_1 & h_2 & \dots & h_n \end{matrix} \mathrm{PrSprt} \begin{matrix} \{g_1\}* & \{g_2\}* & \dots & \{g_n\}* \\ w_1 & w_2 & \dots & w_n \end{matrix}$,

h_j= *solution of Q_j by D_j*, j = 1, 2, …, n,: the notation of the set B_l, l =1,2,…,n, with elements $b_{li_1i_2\dots i_{m_j}}$ =

$Sprt_{x_l}^{\left\{g_{l_{1_{i_1}}}*,\ g_{l_{2_{i_2}}}*,\dots,\ g_{l_{m_j}i_{m_j}}\right\}})_{R_l}$ for any $\left\{l_{i_1},l_{i_2},\dots,l_{i_{m_j}}\right\}$ without repetitions, x_l =

$Sprt_w^{\{K_l\}}$, K_l-set of any $\left\{k_{l_{i_1}}*,k_{l_{i_2}}*,\dots,k_{l_{i_{m_j}}}*\right\}$ without repeating them, l = 1,2,…,n,

$k_{l_{i_j}}$-any digit, i =1,2,…, m_j, R_l= $Sprt_w^{\left\{l_{i_1}+,l_{i_2}+,\dots,l_{i_{m_j}}\right\}}$, R_l is the index of the lower discharge (we choose an index on the scale of discharges):see Table I.1.

Then $\begin{matrix} task\ Q_1 & task\ Q_i & \dots & task\ Q_n \\ h_1 & h_2 & \dots & h_n \end{matrix} \mathrm{PrSprt} \begin{matrix} B_1+ & B_2+ & \dots & B_n+ \\ x_1 & x_2 & \dots & x_n \end{matrix}$ gives the final result of simultaneous multiplication and *solution of Q by D* simultaneously.

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

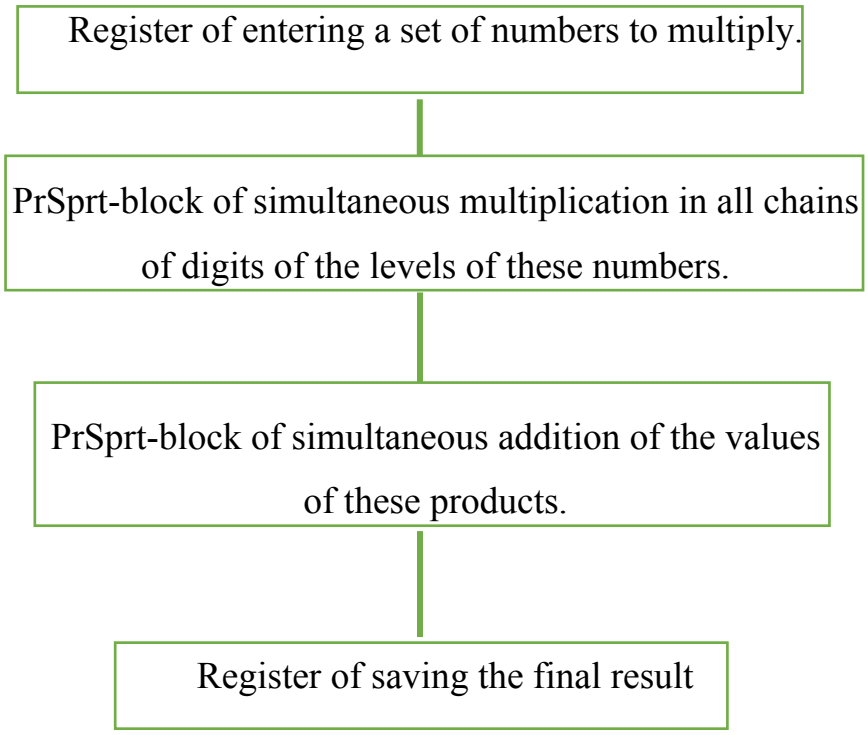


Fig. 3.10. The straightforward functional scheme of the assumed arithmetic-logical device for PrSprt-multiplication.

Remark. 3.10.1. 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 $\{p\} = (p_1,p_2,...,p_n)$ to the variables $\{g\} = (g_1,g_2,...,g_n)$. This is implemented via

   $\underset{h_1}{task\ Q_1}\ \underset{h_2}{task\ Q_2}\ ...\ \underset{h_n}{task\ Q_n}\text{PrSprt}\ g_1 := \underset{p_1}{}\ g_2 := \underset{p_2}{}\ ...\ g_n := \underset{p_n}{}$, h_j= *solution of task* Q_j *by* D_j, j = 1, 2, …, n.

2. Simultaneous checking the set of conditions $\{f\} = (f_1, f_2, ...,f_n)$ for the set of expressions$\{B\} = (B_1, B_2, ...,B_n)$. Implemented via

   $\underset{h_1}{task\ Q_1}\ \underset{h_2}{task\ Q_2}\ ...\ \underset{h_n}{task\ Q_n}\text{PrSprt}\ IF\{B_1 f_1\}\ \underset{x_1}{then}\ \ IF\{B_2 f_2\}\ \underset{x_2}{then}\ \ ...\ \ IF\{B_n f_n\}\ \underset{x_n}{then}$,

   where x_i (i = 1, …, n) can be anything, h_j= *solution of task* Q_j *by* D_j, j = 1, 2, …, n.

3. Similarly for loop operators and others.

PrS_3f– 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 articles.

The ideology of dynamic PrSprt and $PrS_3f(t)$ can be used for programming:

1. The process of simultaneous assignment of the expressions $\{p(t)\} = (p_1(t),p_2(t),...,p_n(t))$ to the variables $\{g(t)\} = (g_1(t),g_2(t),...,g_n(t))$ *and solution of task* Q_j *by* D_j simultaneously is implemented through $\underset{h_1}{task\ Q_1}\ \underset{h_2}{task\ Q_2}\ ...\ \underset{h_n}{task\ Q_n}\text{PrSprt}$

$g_1(t) := \underset{p_1(t)}{}\ g_2(t) := \underset{p_2(t)}{}\ ...\ g_n(t) := \underset{p_n(t)}{}$, h_j= *solution of task* Q_j *by* D_j, j = 1, 2, …, n. .

2. The process of simultaneous check the set of conditions $\{f(t)\} = (f(t)_1, f_{2,}(t) ...,f(t)_n)$ for a set of expressions $\{B(t)\} = (B_1(t), B_2(t), ...,B_n(t))$ *and solution of task* Q_j *by* D_j simultaneously is implemented through

$$\underset{h_1}{task\ Q_1}\ \underset{h_2}{task\ Q_2}\ ...\ \underset{h_n}{task\ Q_n}\ \mathrm{PrSprt}\ IF\{B_1(t)f_1(t)\}\ then\ \underset{x_1(t)}{}\ IF\{B_2(t)f_2(t)\}\ then\ \underset{x_2(t)}{}\ ...\ IF\{B_n(t)f_n(t)\}\ then\ \underset{x_n(t)}{},$$

where x(t) = $(x_1(t),x_2(t),...,x_n(t))$. can be any, h_j= *solution of task* Q_j *by* D_j, j = 1, 2, …, n.

3.Similarly for loop operators and others.

p-analogue of $PrS_3f(t)$– 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 p-analogue of form (1.1) or p-analogue of forms (1.1.1) – (1.4) for more complex operators.

The usage of PrSprt-elements for networks.

A. Galushkin's comprehensive monograph [21] 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 $\begin{matrix} r_1 & r_2 & \dots & r_n \\ \mathrm{u} & \mathrm{u} & \dots & \mathrm{u} \end{matrix} \overleftarrow{PrSprt(t)} \begin{matrix} u & u & \dots & u \\ r_1 & r_2 & \dots & r_n \end{matrix}$, $\mathrm{r_i} =$ *activation with* $g_i(t)$, u = SmnPrSprt, are chosen for necessary tasks. We will not touch on the issues of applications, or network optimization. They are described in detail by Galushkin [21]. 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 –sgn($\mathrm{PrSprt(t)}\begin{matrix} g_1(t)w_1(t) & g_2(t)w_n(t) & \dots & g_n(t)w_n(t) \\ x_1 & x_2 & \dots & x_n \end{matrix}$), $\mathrm{x=(x_1,x_2,\dots,x_n)}$ - mnPrSprt, $\mathrm{w(t)=(\ w_1(t),\ w_2(t)_{,\dots,}w_n(t))}$ – 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 q = $\begin{matrix} D_1 & D_2 & \dots & D_n \\ v & v & \dots & v \end{matrix}\mathrm{PrSprt(t)}\begin{matrix} v & \mathrm{v} & \dots & \mathrm{v} \\ D_1 & D_2 & \dots & D_n \end{matrix}$, v = mnPrSprt, is PrSprt-node of mnPrSprt in range $\mathrm{D = (D_1, D_{2,\dots,} D_n)}$, D- capacity for mnPrSprt node. The third level of mnPrSprt consists of $\begin{matrix} D_1 & D_2 & \dots & D_n \\ q & q & \dots & q \end{matrix}\mathrm{PrSprt(t)}\begin{matrix} q & \mathrm{q} & \dots & \mathrm{q} \\ D_1 & D_2 & \dots & D_n \end{matrix}$ -$\mathrm{PrSprt^2}$- 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 $\mathrm{PrSprt^2}$- nodes of mnPrSprt, but self-level is higher in living organisms, particularly $\mathrm{PrSprt^n}$-, n≥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 - $\begin{matrix} B_1 & B_2 & \dots & B_n \\ \{qy\}_1 & \{qy\}_2 & \dots & \{qy\}_n \end{matrix}$ PrSprt $\begin{matrix} \{ax\}_1 & \{ax\}_2 & \dots & \{ax\}_n \\ B_1 & B_2 & \dots & B_n \end{matrix}$, B_1, B_2, …, 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 3.10.2. A neural network can be thought of as a learnable parallel dynamic operator.

Remark 3.10.3. 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 $\begin{matrix} g_1 & g_2 & \dots & g_n \\ v & v & \dots & v \end{matrix}$PrSprt(t)$\begin{matrix} v & v & \dots & v \\ g_1 & g_2 & \dots & g_n \end{matrix}$, v = mnPrSprt, g_i = $\text{eprogram}_i(t)$, 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 $^{\text{SmnPrSprt}}_{\text{SmnPrSprt}}Sprt^{\text{SmnPrSprt}}_{\text{SmnPrSprt}}$ 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. $\overset{activation}{\underset{\text{SmnPrSprt},f}{}}Sprt^{\text{SmnPrSprt},f}_{activation}$ derives SmnPrSprt to the self-level boundary with target weight f.

It's used an alternating current of above high frequency , 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, S^1e, Set_1.

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 - $\begin{matrix} activation & activation & \dots & activation \\ g_1(t)w_1(t) & g_2(t)w_n(t) & \dots & g_n(t)w_n(t) \end{matrix}\text{PrSprt(t)}\begin{matrix} g_1(t)w_1(t) & g_2(t)w_n(t) & \dots & g_n(t)w_n(t) \\ activation & activation & \dots & activation \end{matrix}$, 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 $\begin{matrix} activation & activation & \dots & activation \\ u_1 & u_2 & \dots & u_n \end{matrix}\text{PrSprt(t)}\begin{matrix} u_1 & u_2 & \dots & u_n \\ activation & activation & \dots & activation \end{matrix}$, $u_i = (\text{UHF AC})_i(t) := Q_i(t)$, we simultaneously

enter the desired set of codes $Q_i(t)$, i = 1, 2, …, 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) S^1e - program operators
4) Set_1- program operators

Here are some of the PrSprt-program operators:

1. Simultaneous assignment of the expressions $\{p\} = (p_1,p_2,...,p_n)$ to the variables $\{g\} = (g_1,g_2,...,g_n)$ and *solution of task* Q_j *by* D_j simultaneously. This is implemented via $\underset{h_1}{task\ Q_1}\ \underset{h_2}{task\ Q_2}\ ...\ \underset{h_n}{task\ Q_n}$PrSprt

$\underset{p_1}{g_1 :=}\ \underset{p_2}{g_2 :=}\ ...\ \underset{p_n}{g_n :=}$, h_j= *solution of* Q_j *by* D_j, j = 1, 2, …, n.

2. Simultaneous checking the set of conditions $\{f\} = (f_1, f_2, ...,f_n)$ for the set of expressions$\{B\} = (B_1, B_2, ...,B_n)$ and *solution of task* Q_j *by* D_j simultaneously.. Implemented via $\underset{h_1}{task\ Q_1}\ \underset{h_2}{task\ Q_2}\ ...\ \underset{h_n}{task\ Q_n}$PrSprt

$\underset{x_1}{IF\{B_1 f_1\}\ then}\ \underset{x_2}{IF\{B_2 f_2\}\ then}\ ...\ \underset{x_n}{IF\{B_n f_n\}\ then}$, where x_i (i = 1, …, n) can be anything, , h_j= *solution of* Q_j *by* D_j, j = 1, 2, …, n.

3. Similarly for loop operators and others.

PRSprt-algorithm Examples:

1) Simultaneous addition and simultaneous parallel multiplication of sets elements $\{g_i\} = \left(g_{i_1}, g_{i_2}, ..., g_{i_{m_j}}\right)$, i = 1, 2, …, n, j =1, 2, …, k (See point 1, 2 in **Math Prself** [2]

2) parallel pattern recognition: PrSprt

$$IF\{q_1 \in image\ archive_1\}\ then \quad IF\{q_2 \in image\ archive_2\}\ then \quad \dots \quad IF\{q_n \in image\ archive_n\}\ then$$
$$Name\ of\ q_1 \qquad\qquad Name\ of\ q_2 \quad \dots \qquad\qquad Name\ of\ q_n$$

The example of PrSprt-program is

$$\text{PrSprt}\,{}^{\text{PrSprt}}{}^{g_1 := \ g_2 := \ \dots \ g_n :=}_{p_1 \quad p_2 \quad \dots \quad p_n}\ {}^{\text{PrSprt}}{}^{IF\{B_1 f_1\}\ then \quad IF\{B_2 f_2\}\ then \quad \dots \quad IF\{B_n f_n\}\ then}_{w_1 \qquad\quad w_2 \quad \dots \qquad\quad w_n}\ \dots\ \text{PrSprt}{}^{w_1 \ w_2 \ \dots \ w_n}_{w_1 \ w_2 \ \dots \ w_n}.$$
$$x_1 \qquad\qquad x_2 \quad \dots \qquad\qquad x_n$$

p-analogue of PrS_3f- software operators will differ only just because aggregates $\{g\},\{p\},\{B\},\{f\}$ will be formed from corresponding PrSprt-program operators in p-analogue of form (1.1) for more complex operators in p-analogue of forms (1.1.1) – (1.4).

For example, $\begin{matrix} g_1\{R\} & g_2\{R\} & \dots & g_n\{R\} \\ \{R\}_1 & \{R\}_2 & \dots & \{R\}_n \end{matrix}\text{PrSprt}\begin{matrix} \{R\}_1 & \{R\}_2 & \dots & \{R\}_n \\ g_1\{R\} & g_2\{R\} & \dots & g_n\{R\} \end{matrix}$ is the capacity in itself of the second type if $g_j\{R\}$ is a program capable of generating $\{R\}_j$.

The example of self-program of the first type is

$$\text{PrSprt}\begin{matrix} \text{PrSprt}{}^{g_1 := \ g_2 := \ \dots \ g_n :=}_{p_1 \quad p_2 \quad \dots \quad p_n}\ {}^{\text{PrSprt}}{}^{IF\{B_1 f_1\}\ then \quad IF\{B_2 f_2\}\ then \quad \dots \quad IF\{B_n f_n\}\ then}_{w_1 \qquad\quad w_2 \quad \dots \qquad\quad w_n}\ \dots\ \text{PrSprt}{}^{w_1 \ w_2 \ \dots \ w_n}_{w_1 \ w_2 \ \dots \ w_n} \\ \text{PrSprt}{}^{g_1 := \ g_2 := \ \dots \ g_n :=}_{p_1 \quad p_2 \quad \dots \quad p_n}\ {}^{\text{PrSprt}}{}^{IF\{B_1 f_1\}\ then \quad IF\{B_2 f_2\}\ then \quad \dots \quad IF\{B_n f_n\}\ then}_{w_1 \qquad\quad w_2 \quad \dots \qquad\quad w_n}\ \dots\ \text{PrSprt}{}^{w_1 \ w_2 \ \dots \ w_n}_{w_1 \ w_2 \ \dots \ w_n} \end{matrix}.$$

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, …, n; j = 1, 2, …, m). For example,

$$\begin{matrix} B_1 & B_2 & \dots & B_n \\ A_1 & A_2 & \dots & A_n \end{matrix}\text{PrSprt}\begin{matrix} A_1 & A_2 & \dots & A_n \\ B_1 & B_2 & \dots & B_n \end{matrix}.$$

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,

$$\begin{matrix} A_1 & A_2 & \dots & A_n \\ A_1 & A_2 & \dots & A_n \end{matrix}\text{PrSprt}\begin{matrix} A_1 & A_2 & \dots & A_n \\ A_1 & A_2 & \dots & A_n \end{matrix}.$$

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 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: PrSprt(t)

$\underset{activation}{(\text{UHF AC})_1(t) := Q_1(t)}$ $\underset{activation}{(\text{UHF AC})_2(t) := Q_2(t)}$ ... $\underset{activation}{(\text{UHF AC})_n(t) := Q_n(t)}$. UHF AC field activation is used.

Consider the dynamic PrSprt and $PrS_3f(t)$ programming:

1. The process of simultaneous assignment of the expressions $\{p(t)\} = (p_1(t),p_2(t),...,p_n(t))$ to the variables $\{g(t)\} = (g_1(t),g_2(t),...,g_n(t))$ and *solution of task* Q_j *by* D_j simultaneously. is implemented through

$\underset{h_1}{task\ Q_1}$ $\underset{h_2}{task\ Q_2}$ ... $\underset{h_n}{task\ Q_n}$ PrSprt $g_1(t) := p_1(t)$ $g_2(t) := p_2(t)$ ... $g_n(t) := p_n(t)$, h_j= *solution of task* Q_j *by* D_j, j = 1, 2, …, n..

2. The process of simultaneous check the set of conditions $\{f(t)\} = (f(t)_1, f_{2,}(t) ...,f(t)_n)$ for a set of expressions $\{B(t)\} = (B_1(t), B_2(t), ...,B_n(t))$ and *solution of task* Q_j *by* D_j simultaneously is implemented through

$\underset{h_1}{task\ Q_1}$ $\underset{h_2}{task\ Q_2}$ ... $\underset{h_n}{task\ Q_n}$ PrSprt $\underset{x_1(t)}{IF\{B_1(t)f_1(t)\}\ then}$ $\underset{x_2(t)}{IF\{B_2(t)f_2(t)\}\ then}$ ... $\underset{x_n(t)}{IF\{B_n(t)f_n(t)\}\ then}$,

where x(t) = $(x_1(t),x_2(t),...,x_n(t))$. can be any, h_j= *solution of task* Q_j *by* D_j, j = 1, 2, …, n.

3.Similarly for loop operators and others.

p-analogue of $PrS_3f(t)$– 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 p-analogue of form (1.1) or p-analogue of 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 $\{p\} = (p_1,p_2,...,p_n)$ from the variables $\{g\} = (g_1,g_2,...,g_n)$. It's implemented through $\underset{=:p_1}{g_1}\ \underset{=:p_2}{g_2}\ \dots\ \underset{=:p_n}{g_n}$ PrSprt.
2. Simultaneous expelling checks the set of conditions $\{f\} = (f_1, f_2, ...,f_n)$ for a set of expressions $\{B\} = (B_1, B_2, ...,B_n)$. It's implemented through

   $IF\{B_1 g_1\}\ \underset{then}{x_1}\quad IF\{B_2 g_2\}\ \underset{then}{x_2}\quad \dots\quad IF\{B_n g_n\}\ \underset{then}{x_n}$ PrSprt, where x_i (i = 1, …, n) can be anything.
3. Similarly for loop operators and others.

$Prt_{S_4 f}$ – 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

$\begin{matrix} C_1 & C_2 & \dots & C_m \\ D_1 & D_2 & \dots & D_m \end{matrix}$PrSrt =

$$\left\{ \sum_{i=1}^{m} Q_i + \begin{matrix} \{\} & \{\} & \dots & \{\} \\ D_1 - D_1 \cap C_1 & D_2 - D_2 \cap C_2 & \dots & D_m - D_m \cap C_m \\ \multicolumn{4}{c}{\sum_{i=1}^{m} (C_i - D_i \cap C_i) - (D_i - D_i \cap C_i)} \end{matrix} \text{PrSrt} \right\},$$ where Q_i is oself-set for $(D_i \cap C_i)$ [14].

Consider the dynamic PrtSpr(t) and $Prt(t)_{S_4 f}$ programming at time t.

1. The process of simultaneous assignment of the expressions $\{p(t)\} = (p_1(t),p_2(t),...,p_n(t))$ to the variables $\{g(t)\} = (g_1(t),g_2(t),...,g_n(t))$ is implemented through $\begin{matrix} g_1(t) := & g_2(t) := & \dots & g_n(t) := \\ p_1(t) & p_2(t) & \dots & p_n(t) \end{matrix}$ PrSprt(t).

2. The process of simultaneous check the set of conditions $\{f(t)\} = (f(t)_1, f_2,(t) ...,f(t)_n)$ for a set of expressions$\{B(t)\} = (B_1(t), B_2(t), ...,B_n(t))$ is implemented through

$$\underset{x_1(t)}{IF\{B_1(t)f_1(t)\}}\ then\quad \underset{x_2(t)\ \ldots}{IF\{B_2(t)f_2(t)\}}\ then\quad \ldots\quad \underset{x_n(t)}{IF\{B_n(t)f_n(t)\}}\ then_{\mathrm{PrSprt(t)},}$$

where x(t) = $(x_1(t),x_2(t),\ldots,x_n(t))$. 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:

$$\begin{matrix} C_1(t) & C_2(t) & \ldots & C_m(t) \\ D_1(t) & D_2(t) & \ldots & D_m(t) \end{matrix}\mathrm{PrSrt(t)} =$$

$$\left\{\begin{matrix} \sum_{i=1}^{m} Q_i(t) + \underset{D_1(t)-D_1(t)\cap C_1(t)}{\{\}} \underset{D_2(t)-D_2(t)\cap C_2(t)}{\{\}} \ldots \underset{D_2(t)-D_2(t)\cap C_2(t)}{\{\}}{}^{\mathrm{PrSrt}} \\ \sum_{i=1}^{m}\left(C_i(t)-D_i(t)\cap C_i(t)\right)-(D_i(t)-D_i(t)\cap C_i(t)) \end{matrix}\right\},$$

where $Q_i(t)$ is oself-set for $\left(D_i(t)\cap C_i(t)\right)$ [11].

Consider PrSrt-program operators

$$Sprt_{t_0}\ \{{}^{q\begin{pmatrix} u & u \\ u & u \end{pmatrix}}_{W_q} Sprt^{E_q}_{q\begin{pmatrix} u & u \\ u & u \end{pmatrix}}{}',\ Sprt^{\{E^{ex}{}_l{}^{d_r}\}}_{d_r\left(E_{in}{}^{l}{}^{d_r}\right)}\}$$ —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:

$$\mathrm{Prf(r,\ u}(E_q)) = Sprt_{t_0}\ \{{}^{q\begin{pmatrix} u & u \\ u & u \end{pmatrix}}_{W_q} Sprt^{E_q}_{q\begin{pmatrix} u & u \\ u & u \end{pmatrix}}{}',\ Sprt^{\{E^{ex}{}_l{}^{d_r}\}}_{d_r\left(E_{in}{}^{l}{}^{d_r}\right)}\}\ (**_{\mathrm{F.1}}).$$

Energy of a living organism of a person:

$$\mathrm{Prhf(r,\ u}(E_q)) = Sprt_{t_0}\ \{{}^{q\begin{pmatrix} u & u \\ u & u \end{pmatrix}}_{W_q} Sprt^{E_q}_{q\begin{pmatrix} u & u \\ u & u \end{pmatrix}}{}',\ Sprt^{\{E^{ex}{}_l{}^{d_r}\}}_{d_{r(self\left(E_{in}{}^{l}{}^{d_r}\right))}}\}\ (***_{\mathrm{F.1}}).$$

$\begin{matrix} u & u \\ u & u \end{matrix}PrSpr^{\begin{matrix} u & u \\ u & u \end{matrix}}_{\begin{matrix} u & u \\ u & u \end{matrix}}$ - 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.

3.11 Program operators PrfSprt, PrtfSpr, fS^{1}e, fSet$_l$ and their pself-type

1. Similarly, for simultaneous parallel multiplication and *solution of task Q_j by D_j* simultaneously:

$$\begin{matrix} task\ Q_1 & task\ Q_i & \dots & task\ Q_n \\ h_1 & h_2 & \dots & h_n \end{matrix} \text{PrfSprt} \begin{matrix} \{g_1\}* & \{g_2\}* & \dots & \{g_n\}* \\ w_1 & w_2 & \dots & w_n \end{matrix}.$$

The notation of the fuzzy set B_l, l =1,2,…,n, with elements

$$b_{li_1i_2\dots i_{m_j}} = Sprt_{x_l}^{\left\{g_{l_{i_1}}*,\ g_{l_{i_2}}*,\dots,\ g_{l_{i_{m_j}}}\right\}})_{R_l} \text{ for any } \left\{l_{i_1},l_{i_2},\dots,l_{i_{m_j}}\right\}$$

without repetitions,$x_l = Sprt_w^{\{K_l\}}$, K_l-set of any $\left\{k_{l_{i_1}}*,k_{l_{i_2}}*,\dots,k_{l_{i_{m_j}}}*\right\}$ without repeating them, l = 1,2,…,n, $k_{l_{i_j}}$-any digit, i =1,2,…, m_j, $R_l= Sprt_w^{\left\{l_{i_1}+,l_{i_2}+,\dots,l_{i_{m_j}}\right\}}$, R_l is the index of the lower discharge (we choose an index on the scale of discharges): see Table I.1.

Then $\begin{matrix} task\ Q_1 & task\ Q_i & \dots & task\ Q_n \\ h_1 & h_2 & \dots & h_n \end{matrix} \text{PrfSprt} \begin{matrix} B_1+ & B_2+ & \dots & B_n+ \\ x_1 & x_2 & \dots & x_n \end{matrix}$ 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 PrfSprt-multiplication:

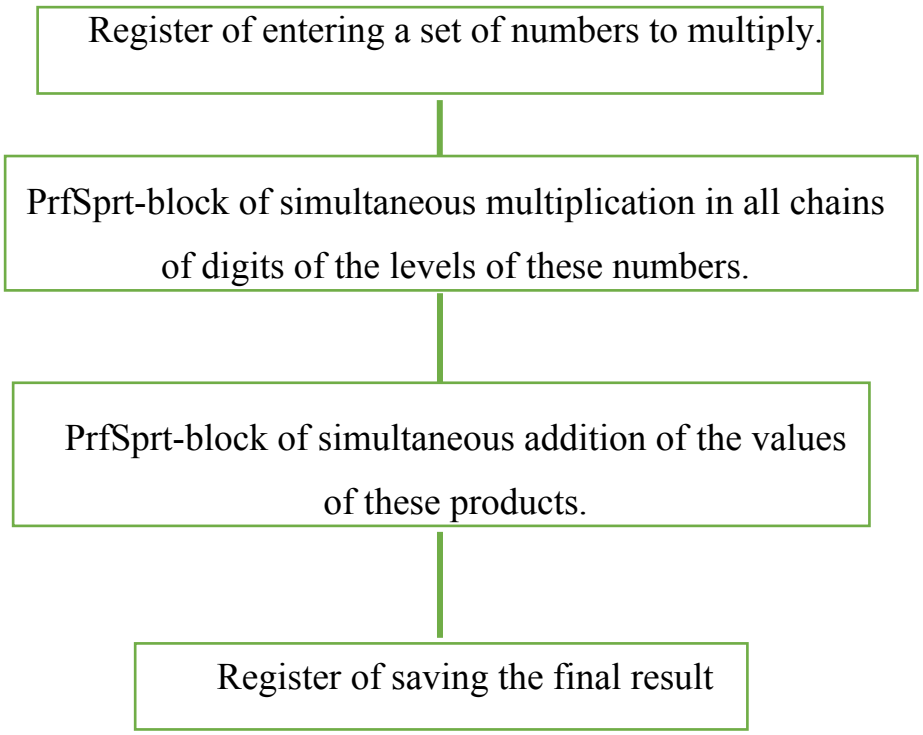


Fig.3.11. The straightforward functional scheme of the assumed arithmetic-logical device for PrfSprt-multiplication.

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 $\tilde{p}=(p_1|\mu_{\tilde{p}}(p_1), p_2|\mu_{\tilde{p}}(p_2), \ldots, p_n|\mu_{\tilde{p}}(p_n))$ to the variables $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \ldots, x_n|\mu_{\tilde{x}}(x_n))$ and *solution of task* Q_j *by* D_j simultaneously. This is implemented via

$$\underset{h_1}{task\ Q_1}\ \underset{h_2}{task\ Q_2}\ \ldots\ \underset{h_n}{task\ Q_n}\ \text{PrfSprt}\ x_1 := p_1\ \ x_2 := p_2\ \ \ldots\ \ x_n := p_n .$$

2. Simultaneous checking the set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2), \ldots, g_n|\mu_{\tilde{g}}(g_n))$ for the set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2), \ldots, B_n|\mu_{\tilde{B}}(B_n))$ and *solution of task* Q_j *by* D_j simultaneously. Implemented via

$$\underset{h_1}{task\ Q_1}\ \underset{h_2}{task\ Q_2}\ \ldots\ \underset{h_n}{task\ Q_n}\ \text{PrSprt}\ \underset{x_1}{IF\{B_1 f_1\}\ then}\ \ \underset{x_2}{IF\{B_2 f_2\}\ then}\ \ \ldots\ \ \underset{x_n}{IF\{B_n f_n\}\ then},$$

where w_i (i = 1, …, n) can be anything, h_j= *solution of task* Q_j *by* D_j, j = 1, 2, …, n.

3. Similarly for loop operators and others.

$PrfS_3f-$ 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 analogues 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 $PrS_3f(t)$ can be used for programming:

1. The process of simultaneous assignment of the expressions $\widetilde{p(t)}=(p_1(t)|\mu_{\widetilde{p(t)}}(p_1(t)), p_2(t)|\mu_{\widetilde{p(t)}}(p_2(t)), \ldots, p_n(t)|\mu_{\widetilde{p(t)}}(p_n(t)))$ to the variables $\widetilde{x(t)}=(x_1(t)|\mu_{\widetilde{x(t)}}(x_1(t)), x_2(t)|\mu_{\tilde{x}(t)}(x_2(t)), \ldots, x_n(t)|\mu_{\widetilde{x(t)}}(x_n(t)))$ and *solution of task* Q_j *by* D_j simultaneously is

implemented through $\overset{task\ Q_1}{h_1}\ \overset{task\ Q_2}{h_2}\ \ldots\ \overset{task\ Q_n}{h_n}$PrfSprt

$x_1(t) := p_1(t)\quad x_2(t) := p_2(t)\quad \ldots\quad x_n(t) := p_n(t)$ . .

2. The process of simultaneous check the set of conditions $\widetilde{g(t)}$=($g_1(t)|\mu_{\widetilde{g(t)}}(g_1(t))$, $g_2(t)|\mu_{\widetilde{g(t)}}(g_2(t)), \ldots, g_n(t)|\mu_{\widetilde{g(t)}}(g_n(t))$) for a set of expressions $\widetilde{B(t)}$=($B_1(t)|\mu_{\widetilde{B(t)}}(B_1(t))$, $B_2(t)|\mu_{\widetilde{B(t)}}(B_2(t)), \ldots, B_n(t)|\mu_{\widetilde{B(t)}}(B_n(t))$) and *solution of task* Q_j *by* D_j simultaneously is implemented through $\overset{task\ Q_1}{h_1}\ \overset{task\ Q_1}{h_2}\ \ldots\ \overset{task\ Q_1}{h_n}$PrfSprt

$IF\{B_1(t)g_1(t)\}\ \overset{then}{w_1(t)}\quad IF\{B_2(t)g_2(t)\}\ \overset{then}{w_2(t)}\quad \ldots\quad IF\{B_n(t)g(t)\}\ \overset{then}{w_n(t)}$, where w(t) = $(w_1(t), w_2(t), \ldots, w_n(t))$. can be any, h_j= *solution of task* Q_j *by* D_j, j = 1, 2, …, n..

3.Similarly for loop operators and others.

$PrfS_3f(t)$– 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 PrfSprt(t) for the above-mentioned programming operators through form (1.1) or forms (1.1.1) - (1.4), (2.1*) and analogues 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 [21] 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 $\begin{matrix} r_1 & r_2 & \dots & r_n \\ u & u & \dots & u \end{matrix}\overleftarrow{PrfSprt(t)}$ $\begin{matrix} u & u & \dots & u \\ r_1 & r_2 & \dots & r_n \end{matrix}$, $r_{i=}$ *activation with* $g_i(t)$, u = SmnPrSprt, are chosen for necessary tasks. We will not touch on the issues of applications, or network optimization. They are described in detail by Galushkin [21]. 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 processinF. More complex executing programs are used for mnPrfSprt nodes. PrfSprt-threshold element –sgn(PrfSprt(t)$\begin{matrix} g_1(t)w_1(t) & g_2(t)w_n(t) & \dots & g_n(t)w_n(t) \\ x_1 & x_2 & \dots & x_n \end{matrix}$),

x=(x_1,x_2,...,x_n) - mnPrfSprt, $\widetilde{W(t)}$=($w_1(t)|\mu_{\widetilde{w(t)}}(w_1(t))$, $w_2(t)|\mu_{\widetilde{w(t)}}(w_2(t))$, ..., $w_n(t)|\mu_{\widetilde{w(t)}}(w_n(t))$).

– 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 q = $\begin{matrix} D_1 & D_2 & \dots & D_n \\ v & v & \dots & v \end{matrix}$PrfSprt(t)$\begin{matrix} v & v & \dots & v \\ D_1 & D_2 & \dots & D_n \end{matrix}$, v = mnPrSprt, – PrfSprt-node of mnPrfSprt in range D = (D_1, D_2, ..., D_n), D- capacity for mnPrfSprt node. The third level of mnPrfSprt consists of $\begin{matrix} D_1 & D_2 & \dots & D_n \\ q & q & \dots & q \end{matrix}$ PrfSprt(t)$\begin{matrix} q & q & \dots & q \\ D_1 & D_2 & \dots & D_n \end{matrix}$ -PrfSprt^2- node of mnPrfSprt in range D, thus D becomes capacity of itself in itself as an element for mnPrfSprt. For our networks, it is sufficient to use PrfSprt^2- nodes of mnPrfSprt, but self-level is higher in living

organisms, particularly PrfSprt^n-, n≥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 PrfSprt

$$\begin{matrix} B_1 & B_2 & \dots & B_n \\ \{qy\}_1 & \{qy\}_2 & \dots & \{qy\}_n \end{matrix} \text{PrfSprt} \begin{matrix} \{ax\}_1 & \{ax\}_2 & \dots & \{ax\}_n \\ B_1 & B_2 & \dots & B_n \end{matrix}$$

, $B_1, B_2, \dots, B_n$ - artificial neurons of type PrfSprt (designation - mnPrfSprt) , $x=(x_1,x_2,\dots,x_n)$ are the fuzzy values of the initial signals, $a=(a_1,a_2,\dots,a_n)$ are the weights of PrfSprt-synapses and the fuzzy values of the output signals $\{qy\}$

Remark 3.9.0. A neural network can be thought of as a learnable parallel dynamic operator.

Remark 3.11.1. 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. mnPrfSprt contains

$$\begin{matrix} g_1 & g_2 & \dots & g_n \\ v & v & \dots & v \end{matrix} \text{PrfSprt(t)} \begin{matrix} v & v & \dots & v \\ g_1 & g_2 & \dots & g_n \end{matrix}.$$

, v = mnPrSprt, g_i = $\text{eprogram}_i(t)$, eprogram −executing program in PrfSprt- OS. PrfSprt-OS (or Prself-OS) is based on PrfSprt-assembly language (or Prself-assembly language), which is based on assembly language through PrfSprt-approach in turn, if the base of elements of PrfSprt-networks is sufficient. The feprograms are in PrfSprt-programming environments (or Prfself-programming environments), but this question and PrfSprt-networks base will be considered in the following monographs. In particular, eprograms may contain PrfSprt- programming operators. In mnPrfSprt 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 interestinF. 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 ${}^{\mathrm{SmnPrfSprt}}_{\mathrm{SmnPrfSprt}}fSprt^{\mathrm{SmnPrfSprt}}_{\mathrm{SmnPrfSprt}}$ 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. $\overset{activation}{\text{SmnPrSprt},f}fSprt^{\text{SmnPrSprt},f}_{activation}$ derives SmnPrfSprt to the self-level boundary with target weight f.

It's used an alternating current of above high frequency , 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, S^1e, Set_1.

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 - $\begin{matrix} activation & activation & \dots & activation \\ g_1(t)w_1(t) & g_2(t)w_n(t) & \dots & g_n(t)w_n(t) \end{matrix}\text{PrfSprt(t)}\begin{matrix} g_1(t)w_1(t) & g_2(t)w_n(t) & \dots & g_n(t)w_n(t) \\ activation & activation & \dots & activation \end{matrix}$, 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 $\overset{activation}{u_1}\ \overset{activation}{u_2}\ \ldots\ \overset{activation}{u_n}\text{PrfSprt(t)}$

$\overset{u_1}{activation}\ \overset{u_2}{activation}\ \ldots\ \overset{u_n}{activation}$, $u_i = (\text{UHF AC})_i(t) := Q_i(t)$, 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) S^{1}e - program operators
4) Set$_1$- program operators

Here are some of the PrfSprt-program operators:

1. Simultaneous assignment of the expressions $\{p\} = (p_1,p_2,...,p_n)$ to the variables $\{g\} = (g_1,g_2,...,g_n)$ and *solution of task* Q_j *by* D_j simultaneously. This is implemented via $\overset{task\ Q_1}{h_1}\ \overset{task\ Q_2}{h_2}\ \ldots\ \overset{task\ Q_n}{h_n}\text{PrfSprt}$

   $g_1 := p_1\ \ g_2 := p_2\ \ \ldots\ \ g_n := p_n$, h_j= *solution of* Q_j *by* D_j, j = 1, 2, …, n..
2. Simultaneous checking the set of conditions $\{f\} = (f_1, f_2, ...,f_n)$ for the set of expressions$\{B\} = (B_1, B_2, ...,B_n)$ and *solution of task* Q_j *by* D_j simultaneously.. Implemented via $\overset{task\ Q_1}{h_1}\ \overset{task\ Q_2}{h_2}\ \ldots\ \overset{task\ Q_n}{h_n}\text{PrfSprt}$

   $IF\{B_1 f_1\}\ \overset{then}{x_1}\ \ IF\{B_2 f_2\}\ \overset{then}{x_2}\ \ldots\ \ IF\{B_n f_n\}\ \overset{then}{x_n}$, where x_i (i = 1, …, n) can be anythinF.
3. Similarly for loop operators and others.

PrfSprt-algorithm Examples:

1. Simultaneous addition and simultaneous parallel multiplication of sets elements $\{g_i\} = \left(g_{i_1}, g_{i_2}, ..., g_{i_{m_j}}\right)$, i = 1, 2, …, n, j =1, 2, …, k (See point 1, 2 in **Math Prself** [2]
2. parallel pattern recognition: PrfSprt
   $IF\{q_1 \in image\ archive_1\}\ then\ Name\ of\ q_1 \quad IF\{q_2 \in image\ archive_2\}\ then\ Name\ of\ q_2 \quad ... \quad IF\{q_n \in image\ archive_n\}\ then\ Name\ of\ q_n$

The example of PrfSprt-program is

$$\text{PrfSprt}\ {}^{\text{PrfSprt}}{}^{g_1 := \ g_2 := \ \dots \ g_n :=}_{p_1 \quad p_2 \quad \dots \quad p_n \quad x_1} \ {}^{\text{PrfSprt}}{}^{IF\{B_1 f_1\}\ then \quad IF\{B_2 f_2\}\ then \quad \dots \quad IF\{B_n f_n\}\ then}_{w_1 \qquad w_2 \quad \dots \qquad w_n \quad x_2 \quad \dots} \ \dots \ \text{PrfSprt}{}^{w_1 \ w_2 \ \dots \ w_n}_{w_1 \ w_2 \ \dots \ w_n \quad x_n}.$$

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, $\begin{matrix} g_1\{R\} & g_2\{R\} & \dots & g_n\{R\} \\ \{R\}_1 & \{R\}_2 & \dots & \{R\}_n \end{matrix} \text{Pr}f\text{Sprt} \begin{matrix} \{R\}_1 & \{R\}_2 & \dots & \{R\}_n \\ g_1\{R\} & g_2\{R\} & \dots & g_n\{R\} \end{matrix}$ is the capacity in itself of the second type if $g_j\{R\}$ is a program capable of generating $\{R\}_j$.

The example of self-program of the first type is

$$\text{PrfSprt}\ {}^{\text{PrfSprt}}{}^{g_1 := \ g_2 := \ \dots \ g_n :=}_{p_1 \quad p_2 \quad \dots \quad p_n} \ {}^{\text{PrfSprt}}{}^{IF\{B_1 f_1\}\ then \quad IF\{B_2 f_2\}\ then \quad \dots \quad IF\{B_n f_n\}\ then}_{w_1 \qquad w_2 \quad \dots \qquad w_n} \ \dots \ \text{PrfSprt}{}^{w_1 \ w_2 \ \dots \ w_n}_{w_1 \ w_2 \ \dots \ w_n}$$
$$\text{PrfSprt}\ {}^{\text{PrfSprt}}{}^{g_1 := \ g_2 := \ \dots \ g_n :=}_{p_1 \quad p_2 \quad \dots \quad p_n} \ {}^{\text{PrfSprt}}{}^{IF\{B_1 f_1\}\ then \quad IF\{B_2 f_2\}\ then \quad \dots \quad IF\{B_n f_n\}\ then}_{w_1 \qquad w_2 \quad \dots \qquad w_n} \ \dots \ \text{PrfSprt}{}^{w_1 \ w_2 \ \dots \ w_n}_{w_1 \ w_2 \ \dots \ w_n}.$$

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, …, n; j = 1, 2, …, m). For example,

$$\begin{matrix} B_1 & B_2 & \dots & B_n \\ A_1 & A_2 & \dots & A_n \end{matrix} \text{PrfSprt} \begin{matrix} A_1 & A_2 & \dots & A_n \\ B_1 & B_2 & \dots & B_n \end{matrix}.$$

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 (F.2.1.1) - (F.2.1.4), where the elements of the sets A_i can be continuous. For example, $\begin{matrix} A_1 & A_2 & \dots & A_n \\ A_1 & A_2 & \dots & A_n \end{matrix} \text{PrfSprt} \begin{matrix} A_1 & A_2 & \dots & A_n \\ A_1 & A_2 & \dots & A_n \end{matrix}$.

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 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:
$\underset{u_1}{activation}\ \underset{u_2}{activation}\ \ldots\ \underset{u_n}{activation}_{\text{PrfSprt(t)}}\ \underset{activation}{u_1}\ \underset{activation}{u_2}\ \ldots\ \underset{activation}{u_n}$, $u_i = (\text{UHF AC})_i(t) := Q_i(t)$, UHF AC field activation is used.

Consider the dynamic PrfSprt and $PrS_3f(t)$ programming:

1. The process of simultaneous assignment of the expressions $\{p(t)\} = (p_1(t),p_2(t),...,p_n(t))$ to the variables $\{g(t)\} = (g_1(t),g_2(t),...,g_n(t))$ and *solution of task* Q_j *by* D_j simultaneously is implemented through

$\underset{h_1}{task\ Q_1}\ \underset{h_2}{task\ Q_2}\ \ldots\ \underset{h_n}{task\ Q_n}_{\text{PrfSprt}}\ g_1(t) := p_1(t)\ \ g_2(t) := p_2(t)\ \ \ldots\ \ g_n(t) := p_n(t)$, h_j = *solution of* Q_j *by* D_j, j = 1, 2, …, n.

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

$\underset{h_1}{task\ Q_1}\ \underset{h_2}{task\ Q_2}\ \ldots\ \underset{h_n}{task\ Q_n}_{\text{PrfSprt}}$

$IF\{B_1(t)f_1(t)\}\ \underset{x_1(t)}{then}\ \ IF\{B_2(t)f_2(t)\}\ \underset{x_2(t)}{then}\ \ \ldots\ \ IF\{B_n(t)f_n(t)\}\ \underset{x_n(t)}{then}$, where x(t) = $(x_1(t),x_2(t),...,x_n(t))$. can be any, h_j= *solution of* Q_j *by* D_j, j = 1, 2, …, n.

3.Similarly for loop operators and others.

p-analogue of $PrS_3f(t)$– software operators will differ only in that the aggregates $\{g(t)\},\{p(t)\},\{B(t)\},\{f(t)\}$ will be formed from corresponding processes PrfSprt(t)

for the above-mentioned programming operators through p-analogue of form (1.1) or p-analogue of 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 programming. Here are some of the PrtSpr - program operators.

1. Simultaneous expelling assignment of the expressions $\{p\} = (p_1,p_2,...,p_n)$ from the variables $\{g\} = (g_1,g_2,...,g_n)$. It's implemented through $\overset{g_1}{=:p_1}\ \overset{g_2}{=:p_2}\ \overset{...}{...}\ \overset{g_n}{=:p_n}$ PrfSprt.
2. Simultaneous expelling checks the set of conditions $\{f\} = (f_1, f_2, ...,f_n)$ for a set of expressions $\{B\} = (B_1, B_2, ...,B_n)$. It's implemented through

   $\text{PrfSprt}_{IF\{B_1f_1\}\ then\ \overset{x_1}{}\ IF\{B_2f_2\}\ then\ \overset{x_2}{}\ ...\ ...\ IF\{B_nf_n\}\ then\ \overset{x_n}{}}$,

   where x_i (i = 1, …, 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

$$\begin{matrix} C_1 & C_2 & ... & C_m \\ D_1 & D_2 & ... & D_m \end{matrix}\text{PrfSrt} =$$

$$\left\{ \sum_{i=1}^{m} Q_i + \begin{matrix} \{\} & \{\} & ... & \{\} \\ D_1 - D_1 \cap C_1 & D_2 - D_2 \cap C_2 & ... & D_m - D_m \cap C_m \\ & \sum_{i=1}^{m} (C_i - D_i \cap C_i) - (D_i - D_i \cap C_i) & & \end{matrix} \text{PrfSrt} \right\}, \text{ where}$$

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 p-assignment of the expressions $\{p(t)\} = (p_1(t),p_2(t),...,p_n(t))$ to the variables $\{g(t)\} = (g_1(t),g_2(t),...,g_n(t))$. is implemented through $\begin{matrix} g_1(t) & g_2(t) := & ... & g_n(t) := \\ := p_1(t) & := p_2(t) & ... & := p_n(t) \end{matrix}$ PrfSprt $\begin{matrix} g_1(t) := & g_2(t) := & ... & g_n(t) := \\ p_1(t) & p_2(t) & ... & p_n(t) \end{matrix}$.

2. 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 PrfSprt(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 PrfSrt(t)-program operator:

$$\begin{matrix} C_1(t) & C_2(t) & ... & C_m(t) \\ D_1(t) & D_2(t) & ... & D_m(t) \end{matrix} \text{PrfSrt(t)} =$$

$$\left\{ \begin{matrix} \sum_{i=1}^{m} Q_i(t) + \begin{matrix} \{\} & \{\} & ... & \{\} \\ D_1(t) - D_1(t) \cap C_1(t) & D_2(t) - D_2(t) \cap C_2(t) & ... & D_2(t) - D_2(t) \cap C_2(t) \end{matrix} \text{PrfSrt} \\ \sum_{i=1}^{m} \left(C_i(t) - D_i(t) \cap C_i(t)\right) - (D_i(t) - D_i(t) \cap C_i(t)) \end{matrix} \right\},$$

where $Q_i(t)$ is oself-set for $\left(D_i(t) \cap C_i(t)\right)$ [14].

Consider PrfSrt- program operators

$fSprt_{t_0}$ $\{ \begin{matrix} q\begin{pmatrix} u & u \\ u & u \end{pmatrix}PrfSpr\begin{matrix} u & u \\ u & u \end{matrix} \\ W_q fSprt_q^{E_q}\begin{pmatrix} u & u \\ u & u \end{pmatrix}PrfSpr\begin{matrix} u & u \\ u & u \end{matrix} \end{matrix}, fSprt_{d_r}^{\{E^{ex}l^{d_r}\}}\left(E_{in}l^{d_r}\right) \}$ —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)) = $fSprt_{t_0}$ $\{ \begin{matrix} q\begin{pmatrix} u & u \\ u & u \end{pmatrix}PrfSpr\begin{matrix} u & u \\ u & u \end{matrix} \\ W_q fSprt_q^{E_q}\begin{pmatrix} u & u \\ u & u \end{pmatrix}PrfSpr\begin{matrix} u & u \\ u & u \end{matrix} \end{matrix}, fSprt_{d_r}^{\{E^{ex}l^{d_r}\}}\left(E_{in}l^{d_r}\right) \}$ (**$_{F.2}$).

Energy of a living organism of a person:

fPrhf(r, u(E_q)) = $fSprt_{t_0}$ $\{ \begin{matrix} q\begin{pmatrix} u & u \\ u & u \end{pmatrix}PrfSpr\begin{matrix} u & u \\ u & u \end{matrix} \\ W_q fSprt_q^{E_q}\begin{pmatrix} u & u \\ u & u \end{pmatrix}PrfSpr\begin{matrix} u & u \\ u & u \end{matrix} \end{matrix}, fSprt_{d_{r(self}\left(E_{in}l^{d_r}\right))}^{\{E^{ex}l^{d_r}\}} \}$ (***$_{F.2}$).

$\begin{matrix} u & u \\ u & u \end{matrix}PrfSpr\begin{matrix} u & u \\ u & u \end{matrix}$ - 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.
($**_{F.2}$), ($***_{F.2}$) can be interpreted as PrfSrt- program operators.

3.12 Program operators PrffSprt, PrtffSpr, S^1e, Set_1 and their pself-type

Threshold element in networks PrffSprt -

$$\begin{matrix} B_1 & B_2 & \dots & B_n \\ \mu_{12} & \mu_{22} & \dots & \mu_{n2} \\ \{qy\}_1 & \{qy\}_2 & \dots & \{qy\}_n \end{matrix} \text{ PrffSprt } \begin{matrix} \{ax\}_1 & \{ax\}_2 & \dots \{ax\}_n \\ \mu_{11} & \mu_{21} & \dots \mu_{n1} \\ B_1 & B_2 & \dots B_n \end{matrix}$$

, $B_1, B_2, \dots, B_n$ - 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 $q=(q_1,q_2,\dots,q_n)$.

1. Similarly, for simultaneous parallel multiplication and

   solution of task Q_j by D_j simultaneously:
   $$\begin{matrix} task\ Q_1 & task\ Q_2 & & task\ Q_n \\ \mu_{21} & \mu_{22} & \dots & \mu_{2n} \\ h_1 & h_2 & \dots & h_n \end{matrix}$$

   $$\text{PrffSprt } \begin{matrix} \{g_1\}* & \{g_2\}* & \dots \{g_n\}* \\ \mu_{11} & \mu_{21} & \dots \mu_{n1} \\ w_1 & w_2 & \dots w_n \end{matrix}$$

   , the notation of the fuzzy set B_l, $l=1,2,\dots,n$, with elements $b_{li_1i_2\dots i_{m_j}} =$

$Sprt_{x_l}^{\left\{g_{l_{1_{i_1}}}*,\ g_{l_{2_{i_2}}}*,\ldots,\ g_{l_{m_j}}{}_{i_{m_j}}\right\}})_{R_l}$ for any $\left\{l_{i_1},l_{i_2},\ldots,l_{i_{m_j}}\right\}$ without repetitions,x_l = $Sprt_w^{\{K_l\}}$,K_l-set of any $\left\{k_{l_{i_1}}*,k_{l_{i_2}}*,\ldots,k_{l_{i_{m_j}}}*\right\}$ without repeating them, l = 1,2,…,n, $k_{l_{i_j}}$-any digit, i =1,2,…, m_j, R_l= $Sprt_w^{\left\{l_{i_1}+,l_{i_2}+,\ldots,l_{i_{m_j}}\right\}}$, R_l is the index of the lower discharge (we choose an index on the scale of discharges): see Table I.1.

	$task\ Q_1$	$task\ Q_2$		$task\ Q_n$		$\{B_1\}+$	$\{B_2\}+$		$\ldots\{B_n\}+$	
Then	μ_{21}	μ_{22}	…	μ_{2n}	PrffSprt	μ_{11}	μ_{21}	…	μ_{n1}	gives
	h_1	h_2	…	h_n		x_1	x_2	…	x_n	

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:

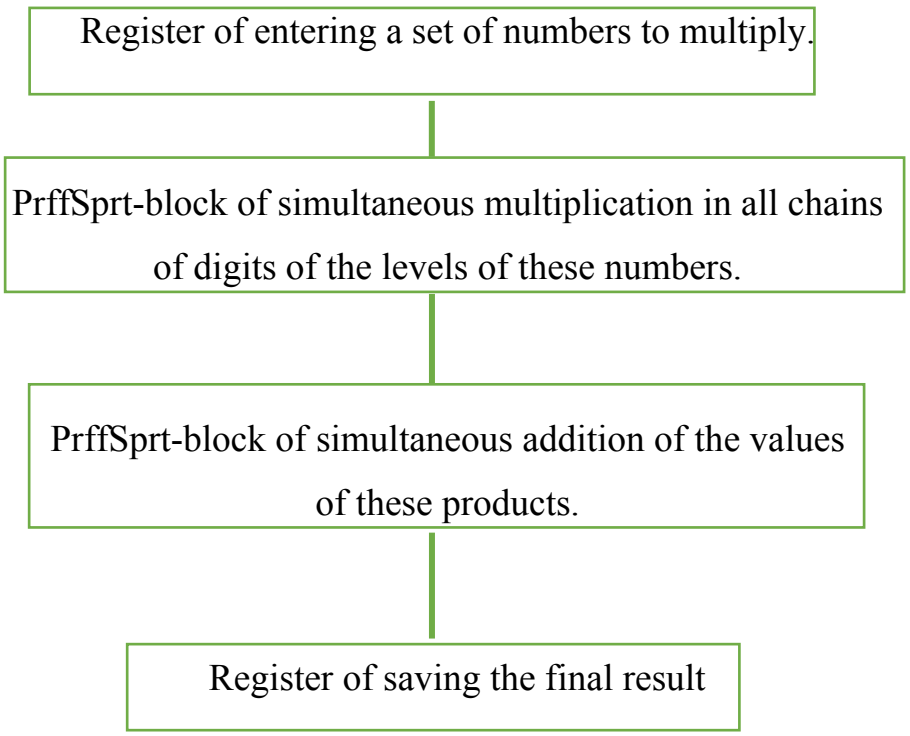


Fig. 3.12. The straightforward functional scheme of the assumed arithmetic-logical device for PrffSprt-multiplication.

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 $\tilde{p}=(p_1|\mu_{\tilde{p}}(p_1), p_2|\mu_{\tilde{p}}(p_2), \ldots, p_n|\mu_{\tilde{p}}(p_n))$ to the variables $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \ldots, x_n|\mu_{\tilde{x}}(x_n))$ and *solution of task* Q_j *by* D_j simultaneously. This is implemented via

$$\begin{matrix} task\ Q_1 & task\ Q_2 & & task\ Q_n & & x_1 := & x_2 := & \ldots x_n := \\ \mu_{21} & \mu_{22} & \ldots & \mu_{2n} & \text{PrffSprt} & \mu_{11} & \mu_{21} & \ldots\ \mu_{n1} \\ h_1 & h_2 & \ldots & h_n & & p_1 & p_2 & \ldots\ p_n \end{matrix}, h_j=$$

solution of task Q_j *by* D_j, j = 1, 2, …, n.

2. Simultaneous fuzzy checking the set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2), \ldots, g_n|\mu_{\tilde{g}}(g_n))$ for the set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2), \ldots, B_n|\mu_{\tilde{B}}(B_n))$ and *solution of task* Q_j *by* D_j simultaneously. Implemented via

$$\begin{matrix} task\ Q_1 & task\ Q_2 & & task\ Q_n & & IF\{B_1g_1\}\ then & IF\{B_2g_2\}\ then & \ldots IF\{B_ng_n\}\ then \\ \mu_{21} & \mu_{22} & \ldots & \mu_{2n} & \text{PrffSprt} & \mu_{11} & \mu_{21} & \ldots\ \mu_{n1} \\ h_1 & h_2 & \ldots & h_n & & w_1 & w_2 & \ldots\ w_n \end{matrix},$$

where w_i (i = 1, …, n) can be anything, h_j= *solution of task* Q_j *by* D_j, j = 1, 2, …, n..

3. Similarly for loop operators and others.

$PrffS_3f-$ 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 analogues 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 $PrffS_3f(t)$ can be used for programming:

1. The process of simultaneous assignment of the expressions $\widetilde{p(t)}=(p_1(t)|\mu_{\widetilde{p(t)}}(p_1(t)), p_2(t)|\mu_{\widetilde{p(t)}}(p_2(t)), \ldots, p_n(t)|\mu_{\widetilde{p(t)}}(p_n(t)))$ to the variables $\widetilde{x(t)}=(x_1(t)|\mu_{\widetilde{x(t)}}(x_1(t)), x_2(t)|\mu_{\tilde{x}(t)}(x_2(t)), \ldots, x_n(t)|\mu_{\widetilde{x(t)}}(x_n(t)))$ and *solution of task* Q_j *by* D_j simultaneously is implemented through

$$\begin{array}{cccccc} task\ Q_1 & task\ Q_2 & & task\ Q_n & & x_1(t) := \ x_2(t) := \ \ldots x_n(t) := \\ \mu_{21} & \mu_{22} & \cdots & \mu_{2n} & \text{PrffSprt(t)} & \mu_{11} \quad \mu_{21} \quad \ldots \quad \mu_{n1} \\ h_1 & h_2 & \ldots & h_n & & p_1(t) \quad p_2(t) \quad \ldots \ p_n(t) \end{array}, h_j=$$

solution of task Q_j *by* D_j with measure of fuzziness μ_{2j}, j = 1, 2, …, n.

2. The process of simultaneous check the set of conditions $\widetilde{g(t)}=(g_1(t)|\mu_{\widetilde{g(t)}}(g_1(t)), g_2(t)|\mu_{\widetilde{g(t)}}(g_2(t)), \ldots, g_n(t)|\mu_{\widetilde{g(t)}}(g_n(t)))$ for a set of expressions $\widetilde{B(t)}=(B_1(t)|\mu_{\widetilde{B(t)}}(B_1(t)), B_2(t)|\mu_{\tilde{B}(t)}(B_2(t)), \ldots, B_n(t)|\mu_{\widetilde{B(t)}}(B_n(t)))$ and *solution of task* Q_j *by* D_j simultaneously is implemented through

$$\begin{array}{cccccc} task\ Q_1 & task\ Q_2 & & task\ Q_n & & IF\left\{B_1(t)g_1(t)\right\}\ then \quad IF\left\{B_2(t)g_2(t)\right\}\ then \quad \ldots IF\left\{B_n(t)g_n(t)\right\}\ then \\ \mu_{21} & \mu_{22} & \cdots & \mu_{2n} & \text{PrffSprt(t)} & \mu_{11} \quad \mu_{21} \quad \ldots \quad \mu_{n1} \\ h_1 & h_2 & \ldots & h_n & & w_1(t) \quad w_2(t) \quad \ldots \quad w_n(t) \end{array}$$

here $w(t) = (w_1(t), w_2(t), \ldots, w_n(t))$. can be any, h_j= *solution of task* Q_j *by* D_j with measure of fuzziness μ_{2j}, j = 1, 2, …, n.

3.Similarly for loop operators and others.

p-analogue of $PrffS_3f(t)$– software operators will differ only in that the aggregates $\left\{\widetilde{g(t)}\right\}, \left\{\widetilde{p(t)}\right\}, \left\{\widetilde{B(t)}\right\}, \left\{\widetilde{x(t)}\right\}$ will be formed from corresponding processes PrffSprt(t) for the above-mentioned programming operators through p-

analogue of form (1.1) or p-analogue of forms (1.1.1) - (.1.4), (2.1*) and analogues 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 [21] 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 all matrix rows simultaneously. PrffSprt-translation is carried out by inversion. In this case, PrffSelf-coding and PrffSelf-translation will be more stable. The target weights

$$g_i(t) \text{ in } \begin{matrix} r_1 & r_2 & \dots & r_n \\ \mu_{11} & \mu_{12} & & \mu_{1n} \\ u & u & \dots & u \end{matrix} \xleftarrow{PrffSprt(t)} \begin{matrix} u & u & \dots & u \\ \mu_{21} & \mu_{22} & & \mu_{2n} \\ r_1 & r_2 & \dots & r_n \end{matrix}, \text{ r}_\text{i} = \textit{activation with } g_i(t),$$

u = SmnPrSprt, are chosen for necessary tasks. We will not touch on the issues of applications, or network optimization. They are described in detail by Galushkin [21]. 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 PrffSprt (designation - mnPrffSprt) for simple information processinF. More complex executing programs are used for mnPrffSprt nodes.

PrffSprt-threshold element –sgn(PrffSprt(t) $\begin{matrix} g_1(t)w_1(t) & g_2(t)w_2(t) & \dots g_n(t)w_n(t) \\ \mu_{11} & \mu_{21} & \dots \ \mu_{n1} \\ x_1 & x_2 & \dots \ x_n \end{matrix}$),

x=(x_1,x_2,…,x_n) - mnPrffSprt, $\widetilde{W}(t)$=($w_1(t)|\mu_{\widetilde{w}(t)}(w_1(t))$, $w_2(t)|\mu_{\widetilde{w}(t)}(w_2(t))$, …, $w_n(t)|\mu_{\widetilde{w}(t)}(w_n(t))$).

– source signals values, $\{g(t)\} = (g_1(t),g_2(t),\dots,g_n(t))$ – PrffSprt-synapses weights. The first level of mnPrffSprt consists of simple mnPrffSprt. The second level of mnPrffSprt consists of q = $\begin{matrix} D_1 & D_2 & \dots & D_n \\ \mu_{11} & \mu_{12} & & \mu_{1n} \\ v & v & \dots & v \end{matrix}$ $\underleftarrow{PrffSprt(t)}$ $\begin{matrix} v & v & \dots & v \\ \mu_{21} & \mu_{22} & & \mu_{2n} \\ D_1 & D_2 & \dots & D_n \end{matrix}$, v = mnPrSprt, – PrffSprt-node of mnPrffSprt in range D = (D_1, D_2,…, D_n), D- fcapacity for mnPrffSprt node. The third level of mnPrffSprt consists of

$\begin{matrix} D_1 & D_2 & \dots & D_n \\ \mu_{11} & \mu_{12} & & \mu_{1n} \\ q & q & \dots & q \end{matrix}$ $\underleftarrow{PrffSprt(t)}$ $\begin{matrix} q & q & \dots & q \\ \mu_{21} & \mu_{22} & & \mu_{2n} \\ D_1 & D_2 & \dots & D_n \end{matrix}$ -PrffSprt^2- node of mnPrffSprt in range D, thus D becomes fcapacity of itself in itself as an element for mnPrffSprt. For our networks, it is sufficient to use PrffSprt^2- nodes of mnPrffSprt, but self-level is higher in living organisms, particularly PrffSprt^n-, n≥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 PrffSprt - $\begin{matrix} B_1 & B_2 & \dots & B_n \\ \mu_{12} & \mu_{22} & \dots & \mu_{n2} \\ \{qy\}_1 & \{qy\}_2 & \dots & \{qy\}_n \end{matrix}$ PrffSprt $\begin{matrix} \{ax\}_1 & \{ax\}_2 & \dots \{ax\}_n \\ \mu_{11} & \mu_{21} & \dots \ \mu_{n1} \\ B_1 & B_2 & \dots \ B_n \end{matrix}$, B_1, B_2, …, B_n - artificial neurons of type PrffSprt (designation - mnPrffSprt) , $\tilde{x}$=($x_1|\mu_{\tilde{x}}(x_1)$, $x_2|\mu_{\tilde{x}}(x_2)|$,…,$x_n|\mu_{\tilde{x}}(x_n)|$) are the fuzzy values of the initial signals, a=(a_1,a_2,…,a_n) are the weights of PrffSprt-synapses and $\tilde{y}$=($y_1|\mu_{\tilde{y}}(y_1)$, $y_2|\mu_{\tilde{y}}(y_2)|$, …, $y_n|\mu_{\tilde{y}}(y_n)|$) are the fuzzy values of the output signals $\{qy\}$ with weights q=(q_1,q_2,…,q_n).

Remark 3.12.0. A neural network can be thought of as a learnable parallel fuzzy dynamic operator.

Remark 3.12.1. 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 contains $\begin{matrix} q & q & \dots & q \\ \mu_{21} & \mu_{22} & & \mu_{2n} \\ w_1(t) & w_2(t) & \dots & w_n(t) \end{matrix}$ PrffSprt(t) $\begin{matrix} w_1(t) & w_2(t) & \dots w_n(t) \\ \mu_{11} & \mu_{21} & \dots \ \mu_{n1} \\ \mathrm{q} & \mathrm{q} & \dots \ \mathrm{q} \end{matrix}$, v = mnPrSprt, $w_j(t)$ = ffeprogram$_j(t)$ –executing program in PrffSprt- OS, j = 1, 2, …, n. 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 interestinF. 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 $\underset{\text{SmnPrffSprt}}{\text{SmnPrffSprt}} fSprt^{\text{SmnPrffSprt}}_{\text{SmnPrffSprt}}$ 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 ffeprograms 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. $\underset{\text{SmnPrffSprt},f}{activation} fSprt^{\text{SmnPrffSprt},f}_{activation}$ 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 -

$$\begin{matrix} activation & activation & & activation \\ \mu_{21} & \mu_{22} & \cdots & \mu_{2n} \\ g_1(t)w_1(t) & g_2(t)w_2(t) & \ldots & g_n(t)w_n(t) \end{matrix} \xleftarrow{\text{PrffSprt(t)}} \begin{matrix} g_1(t)w_1(t) & g_2(t)w_2(t) & \ldots g_n(t)w_n(t) \\ \mu_{11} & \mu_{21} & \ldots \mu_{n1} \\ activation & activation & \ldots activation \end{matrix}$$

, it will be possible to obtain the execution of the parallel fuzzy actions $(g_1(t), g_2(t), \ldots, g_n(t))$ with the desired target weights w(t)=($w_1(t), w_2(t), \ldots, w_n(t)$). 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

$$\begin{matrix} activation & activation & & activation \\ \mu_{21} & \mu_{22} & \cdots & \mu_{2n} \\ h_1(t) & h_2(t) & \ldots & h_n(t) \end{matrix} \xleftarrow{\text{PrffSprt(t)}} \begin{matrix} h_1(t) & h_2(t) & \ldots & h_n(t) \\ \mu_{11} & \mu_{21} & \ldots & \mu_{n1} \\ activation & activation & \ldots activation & \end{matrix}$$

, $h_i(t) = (\text{UHF AC})_i(t) := Q_i(t)$, we simultaneously enter the desired fuzzy set of codes

$Q_i(t)$, i = 1, 2, …, 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.

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: PrffSprt

$$\begin{array}{cccc} IF\left\{q_1 \in image\ archive_1\right\}\ then & IF\left\{q_2 \in image\ archive_2\right\}\ then & ...IF\left\{q_n \in image\ archive_n\right\}\ then \\ \mu_{11} & \mu_{21} & ... & \mu_{n1} \\ Name\ of\ q_1 & Name\ of\ q_2 & ... & Name\ of\ q_n \end{array}$$

The example of PrffSprt-program is

$$\begin{array}{rcccccccccccc} & x_1 := & x_2 := & ...x_n := & & IF\{B_1 g_1\}\ then & IF\{B_2 g_2\}\ then & ...IF\{B_n g_n\}\ then & & w_1 & w_2 & ...w_n \\ \text{PrffSprt} & \mu_{11} & \mu_{21} & ... \mu_{n1} & \text{PrffSprt} & \mu_{11} & \mu_{21} & ... \mu_{n1} & ...\text{PrffSprt} & \mu_{11} & \mu_{21} & ...\mu_{n1}, \\ \text{PrffSprt} & p_1 & p_2 & ... p_n & & w_1 & w_2 & ... w_n & & w_1 & w_2 & ...w_n \\ & \mu_{11} & & & & \mu_{21} & & & ... & \mu_{n1} & & \\ & r_1 & & & & r_2 & & & ... & r_n & & \end{array}$$

$PrffS_3f-$ 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 analogues of forms (1.1.1) - (1.4) by type (2.1*).

For example, $fSprt^{\{R\}}_{g\{R\}}$ 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

$$\begin{array}{rcccccccccccc} & x_1 := & x_2 := & ...x_n := & & IF\{B_1 g_1\}\ then & IF\{B_2 g_2\}\ then & ...IF\{B_n g_n\}\ then & & w_1 & w_2 & ...w_n \\ \text{PrffSprt} & \mu_{11} & \mu_{21} & ... \mu_{n1} & \text{PrffSprt} & \mu_{11} & \mu_{21} & ... \mu_{n1} & ...\text{PrffSprt} & \mu_{11} & \mu_{21} & ...\mu_{n1}, \\ & p_1 & p_2 & ... p_n & & w_1 & w_2 & ... w_n & & w_1 & w_2 & ...w_n \\ \text{PrffSprt} & \mu_{11} & & & & \mu_{21} & & & ... & \mu_{n1} & & . \\ & x_1 := & x_2 := & ...x_n := & & IF\{B_1 g_1\}\ then & IF\{B_2 g_2\}\ then & ...IF\{B_n g_n\}\ then & & w_1 & w_2 & ...w_n \\ \text{PrffSprt} & \mu_{11} & \mu_{21} & ... \mu_{n1} & \text{PrffSprt} & \mu_{11} & \mu_{21} & ... \mu_{n1} & ... \text{PrffSprt} & \mu_{11} & \mu_{21} & ...\mu_{n1} \\ & p_1 & p_2 & ... p_n & & w_1 & w_2 & ... w_n & & w_1 & w_2 & ...w_n \end{array}$$

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, …, n; j = 1, 2, …, m). For example, $\begin{array}{cccc} B_1 & B_2 & ...B_n & \\ \mu_{12} & \mu_{22} & ...\mu_{n2} & \text{PrffSprt} \\ A_1 & A_2 & ...A_n & \end{array} \begin{array}{cccc} A_1 & A_2 & ...A_n \\ \mu_{11} & \mu_{21} & ...\mu_{n1}. \\ B_1 & B_2 & ...B_n \end{array}$

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 continuous. For example, $\begin{matrix} A_1 & A_2 & \dots A_n \\ \mu_{12} & \mu_{22} & \dots \mu_{n2} \\ A_1 & A_2 & \dots A_n \end{matrix}$ PrffSprt $\begin{matrix} A_1 & A_2 & \dots A_n \\ \mu_{11} & \mu_{21} & \dots \mu_{n1}. \\ A_1 & A_2 & \dots A_n \end{matrix}$

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 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:

$\begin{matrix} activation & activation & & activation \\ \mu_{21} & \mu_{22} & \dots & \mu_{2n} \\ h_1(t) & h_2(t) & \dots & h_n(t) \end{matrix}$ $\overleftarrow{\text{PrffSprt(t)}}$ $\begin{matrix} h_1(t) & h_2(t) & \dots & h_n(t) \\ \mu_{11} & \mu_{21} & \dots & \mu_{n1} \\ activation & activation & \dots & activation \end{matrix}$, $h_i(t) = (\text{UHF AC})_i(t) := Q_i(t)$, UHF AC field activation is used.

Consider PrfftSpr-program operators. The ideology of PrftSpr and $Prft_{S_4f}$ is parallel analogue of ft_{S_4f} [11] can be used for programminF. Here are some of the PrftSpr -program operators.

1. Simultaneous fuzzy expelling assignment of the fuzzy expressions $\tilde{p}$ $=(p_1|\mu_{\tilde{p}}(p_1), p_2|\mu_{\tilde{p}}(p_2), \dots, p_n|\mu_{\tilde{p}}(p_n))$ from the variables $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \dots, x_n|\mu_{\tilde{x}}(x_n))$. It's implemented through $\begin{matrix} x_1 := & x_2 := & \dots x_n := \\ \mu_{11} & \mu_{21} & \dots \mu_{n1} \\ p_1 & p_2 & \dots p_n \end{matrix}$ PrffSprt.
2. 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 analogues of forms (1.1.1) - (1.4) by type (2.1*).

Consider hierarchical PrfftSpr-program operator

$$\begin{matrix} C_1(t) & C_2(t) & \dots C_m(t) \\ \mu_{12} & \mu_{22} & \dots \ \mu_{m2} \\ D_1(t) & D_2(t) & \dots D_m(t) \end{matrix} \text{ PrffSrt}_2\text{(t)} =$$

$$\left\{ \begin{matrix} \sum_{i=1}^{m} Q_i(t) + & \{\} & \{\ \} & \dots & \{\} & \\ & \mu_{12} & \mu_{22} & \dots & \mu_{m2} & \text{PrffSrt(t)} \\ & D_1(t) - D_1(t) \cap C_1(t) & D_2(t) - D_2(t) \cap C_2(t) & \dots & D_m(t) - D_2(t) \cap C_m(t) & \\ & \sum_{i=1}^{m} (\, C_i(t) - D_i(t) \cap C_i(t)) - (D_i(t) - D_i(t) \cap C_i(t)) & & & & \end{matrix} \right\}$$

, 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.

1. The process of simultaneous assignment of the expressions $\widetilde{p(t)}$=($p_1(t)|\mu_{\widetilde{p(t)}}$ ($p_1(t)$), $p_2(t)|\mu_{\widetilde{p(t)}}(p_2(t))$, ..., $p_n(t)|\mu_{\widetilde{p(t)}}(p_n(t))$ to the variables $\widetilde{x(t)}$=($x_1(t)|\mu_{\widetilde{x(t)}}(x_1(t))$, $x_2(t)|\mu_{\tilde{x}(t)}(x_2(t))$, ..., $x_n(t)|\mu_{\widetilde{x(t)}}(x_n(t))$ is implemented through

$$\begin{matrix} x_1(t) & x_2(t) & \dots & x_n(t) \\ \mu_{11} & \mu_{21} & \dots & \mu_{n1} \\ := p_1(t) & := p_2(t) & \dots := & p_n(t) \end{matrix} \text{ PrffSprt(t).}$$

2. Similarly for loop operators and others.

p-analogue of $Prfft(t)_{S_4f}$ – software operators will differ only in that the aggregates $\widetilde{x(t)}$,$\widetilde{p(t)}$, $\widetilde{B(t)}$,$\widetilde{g(t)}$ will be formed from corresponding processes PrffSprt(t) for the above-mentioned programming operators through p-analogue of form (1.1) or p-analogue of forms (1.1.1) – (1.4) for more complex operators, (2.1*) and analogues of forms (1.1.1) - (1.4) by type (2.1*).

Consider PrffSrt(t)- program operators

$$\{ \begin{matrix} q \begin{pmatrix} u & u & u & u \\ \mu_1 & \mu_2 PrffSpr \mu_1 & \mu_2 \\ u & u & u & u \end{pmatrix} \\ W_q \end{matrix} fSprt_q^{E_q} \begin{pmatrix} u & u & u & u \\ \mu_1 & \mu_2 PrffSpr \mu_1 & \mu_2 \\ u & u & u & u \end{pmatrix}, \ fSprt_{d_r\left(E_{in} l^{d_r}\right)}^{\{E^{ex_l d_r}\}} \}$$

$fSprt_{t_0}$. —program structure example, where the assemblage point d_r is the cursor, it is quite complex ffself—program.

Remark. Energy with measure of fuzziness μ_1, μ_2 of a living organism other than humans:

fPrff(r, u(E_q), μ_1, μ_2) =

$$\{ {}^{q}\begin{pmatrix} u & u & u & u \\ \mu_1 & \mu_2 Prff Spr \mu_1 & \mu_2 \\ u & u & u & u \end{pmatrix} {}^{E_q}_{W_q} Sprt_q \begin{pmatrix} u & u & u & u \\ \mu_1 & \mu_2 Prff Spr \mu_1 & \mu_2 \\ u & u & u & u \end{pmatrix}', \; fSprt^{\{E^{ex}l^{d_r}\}}_{d_r\left(E_{in}l^{d_r}\right)} \}$$

$$fSprt_{t_0} \qquad (**_{F.3}).$$

Energy with measure of fuzziness μ_1, μ_2 of a living organism of a person:

fPrhff(r, u(E_q), μ_1, μ_2) =

$$\{ {}^{q}\begin{pmatrix} u & u & u & u \\ \mu_1 & \mu_2 Prff Spr \mu_1 & \mu_2 \\ u & u & u & u \end{pmatrix} {}^{E_q}_{W_q} Sprt_q \begin{pmatrix} u & u & u & u \\ \mu_1 & \mu_2 Prff Spr \mu_1 & \mu_2 \\ u & u & u & u \end{pmatrix}', \; fSprt^{\{E^{ex}l^{d_r}\}}_{d_r(self\left(E_{in}l^{d_r}\right))} \}$$

$$fSprt_{t_0} \qquad (***_{F.3}).$$

$\begin{matrix} u & u & u & u \\ \mu_1 & \mu_2 Prff Spr \mu_1 & \mu_2 \\ u & u & u & u \end{matrix}$ -internal energy with measure of fuzziness μ_1, μ_2 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.

($**_{F.3}$), ($***_{F.3}$) can be interpreted as PrfSrt- program operators.

3.13 Program operators PrSCprt, PrtSCpr and their pself-type

Here are some of the PrSCprt programming operators.

1. Simultaneous assignment of the expressions $\tilde{p}=(p_1|\mu_{\tilde{p}}(p_1), p_2|\mu_{\tilde{p}}(p_2), \ldots, p_n|\mu_{\tilde{p}}(p_n))$ to the variables $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \ldots, x_n|\mu_{\tilde{x}}(x_n))$ with types of assignment g_{j1}, and expelling of the expressions $\tilde{r}=(r_1|\mu_{\tilde{r}}(r_1), r_2|\mu_{\tilde{r}}(r_2)|, \ldots, r_n|\mu_{\tilde{r}}(r_n))$ from the variables $\tilde{y}=(y_1|\mu_{\tilde{y}}(y_1), y_2|\mu_{\tilde{y}}(y_2)|, \ldots, y_n|\mu_{\tilde{y}}(y_n))$ with types of expelling g_{2j} simultaneously, j = 1, 2, …, n. This is implemented via

$$\begin{matrix} y_1 & y_2 & \cdots & y_n \\ g_{21} & g_{22} & \cdots & g_{2n} \\ := r_2 & := r_2 & \ldots := & r_n \end{matrix} \text{ PrSCprt } \begin{matrix} x_1 := & x_2 := & \ldots x_n := \\ g_{11} & g_{21} & \cdots\ g_{n1} \\ p_1 & p_2 & \cdots\ p_n \end{matrix} .$$

2. Simultaneous p-checking the set of conditions $\tilde{w}=(w_1|\mu_{\tilde{w}}(w_1), w_2|\mu_{\tilde{w}}(w_2), \ldots, w_n|\mu_{\tilde{w}}(w_n))$ for the set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2), \ldots, B_n|\mu_{\tilde{B}}(B_n))$. Implemented via

$$\begin{matrix} u_1 & u_2 & \cdots & u_n \\ g_{11} & g_{21} & \cdots & g_{n1} \\ IF\{B_1w_1\}\ then\ Q_1 & IF\{B_2w_2\}\ then\ Q_2 & \ldots IF\{B_nw_n\}\ then\ Q_n \end{matrix} \text{ PrSCprt } \begin{matrix} IF\{B_1w_1\}\ then\ Q_1 & IF\{B_2w_2\}\ then\ Q_2 & \ldots IF\{B_nw_n\}\ then\ Q_n \\ g_{11} & g_{21} & \cdots\ g_{n1} \\ u_1 & u_2 & \cdots\ u_n \end{matrix}$$

, or

$$\begin{matrix} u_1 & u_2 & \cdots & u_n \\ g_{11} & g_{21} & \cdots & g_{n1} \\ IF\{B_1w_1\}\ then\ Q_1 & IF\{B_2w_2\}\ then\ Q_2 & \ldots IF\{B_nw_n\}\ then\ Q_n \end{matrix} \overleftarrow{\text{PrSCprt}} \begin{matrix} IF\{B_1w_1\}\ then\ Q_1 & IF\{B_2w_2\}\ then\ Q_2 & \ldots IF\{B_nw_n\}\ then\ Q_n \\ g_{11} & g_{21} & \cdots\ g_{n1} \\ u_1 & u_2 & \cdots\ u_n \end{matrix}$$

for ordered case, where u_i (i = 1, …, n) can be anything.

3. Similarly for loop operators and others.

p-analogue of PrS_3Cf- software operators will differ only in that the aggregates $\{\tilde{w}\},\{\tilde{p}\},\{\tilde{B}\},\{\tilde{x}\}$ will be formed from the corresponding PrSCprt program operators in p-analogue of form (1.1) and for more complex operators in p-analogue of forms (1.1.1) –(1.4), (2.1*) and analogues 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.

About dynamic PrSCprt and $PrS_3Cf(t)$ programming.

The ideology of dynamic PrSCprt and $PrS_3Cf(t)$ can be used for programming:

1. The process of simultaneous assignment of the expressions $p\widetilde{(t)}=(p_1(t)|\mu_{p\widetilde{(t)}}(p_1(t)), p_2(t)|\mu_{p\widetilde{(t)}}(p_2(t)), \ldots, p_n(t)|\mu_{p\widetilde{(t)}}(p_n(t)))$ to the variables $x\widetilde{(t)}=(x_1(t)|\mu_{x\widetilde{(t)}}(x_1(t)),$

$x_2(t)|\mu_{\tilde{x}(t)}(x_2(t))_{,\ldots,}\, x_n(t)|\mu_{\widetilde{x(t)}}(x_n(t)))$ with types of assignment g_{2j} and expelling of the expressions $\tilde{r}=(r_1|\mu_{\tilde{r}}(r_1), r_2|\mu_{\tilde{r}}(r_2)|_{,\ldots,}\, r_n|\mu_{\tilde{r}}(r_n))$ from the variables $\tilde{y}=(y_1|\mu_{\tilde{y}}(y_1), y_2|\mu_{\tilde{y}}(y_2)|_{,\ldots,} y_n|\mu_{\tilde{y}}(y_n))$ with types of expelling g_{2j} simultaneously, j = 1, 2, …, n.

simultaneously is implemented through

$$\begin{matrix} y_1(t) & y_2(t) & \ldots & y_n(t) \\ g_{11} & g_{21} & \ldots & g_{n1} \\ := r_1(t) & := r_2(t) & \ldots := & r_n(t) \end{matrix} \text{PrSCprt(t)} \begin{matrix} x_1(t) := & x_2(t) := & \ldots x_n(t) := \\ g_{11} & g_{21} & \ldots \quad g_{n1} \\ p_1(t) & p_2(t) & \ldots \; p_n(t) \end{matrix} .$$

2. The process of simultaneous p-check the set of conditions $\widetilde{A(t)}=(A_1(t)|\mu_{\widetilde{A(t)}}(A_1(t)), A_2(t)|\mu_{\tilde{A}(t)}(A_2(t))_{,\ldots,}\, A_n(t)|\mu_{\widetilde{A(t)}}(A_n(t)))$ for a set of expressions $\widetilde{B(t)}=(B_1(t)|\mu_{\widetilde{B(t)}}(B_1(t)), B_2(t)|\mu_{\tilde{B}(t)}(B_2(t))_{,\ldots,}\, B_n(t)|\mu_{\widetilde{B(t)}}(B_n(t)))$ is implemented through

$$\begin{matrix} w_1(t) & w_2(t) & \ldots & w_n(t) \\ \mu_{11} & \mu_{21} & \ldots & \mu_{n1} \\ IF\{B_1(t)\,A_1(t)\}\ then & IF\{B_2(t)\,A_2(t)\}\ then & \ldots IF\{B_n(t)\,A_n(t)\}\ then \end{matrix} \text{PrSCprt(t)} \begin{matrix} IF\{B_1(t)A_1(t)\}\ then & IF\{B_2(t)A_2(t)\}\ then & \ldots IF\{B_n(t)A_n(t)\}\ then \\ g_{11} & g_{21} & \ldots \quad g_{n1} \\ w_1(t) & w_2(t) & \ldots \; w_n(t) \end{matrix}$$

or

$$\begin{matrix} w_1(t) & w_2(t) & \ldots & w_n(t) \\ \mu_{11} & \mu_{21} & \ldots & \mu_{n1} \\ IF\{B_1(t)\,A_1(t)\}\ then & IF\{B_2(t)\,A_2(t)\}\ then & \ldots IF\{B_n(t)\,A_n(t)\}\ then \end{matrix} \overleftarrow{\text{PrSCprt(t)}} \begin{matrix} IF\{B_1(t)A_1(t)\}\ then & IF\{B_2(t)A_2(t)\}\ then & \ldots IF\{B_n(t)A_n(t)\}\ then \\ g_{11} & g_{21} & \ldots \quad g_{n1} \\ w_1(t) & w_2(t) & \ldots \; w_n(t) \end{matrix}$$

for ordered case, here $w(t) = (w_1(t), w_2(t), \ldots, w_n(t))$. can be any.

3.Similarly for loop operators and others.

$PrS_3Cf(t)$– software operators will differ only in that the aggregates $\{\widetilde{A(t)}\}, \{\widetilde{p(t)}\}, \{\widetilde{B(t)}\}, \{\widetilde{x(t)}\}$ will be formed from corresponding processes PrSCprt(t) for the above-mentioned programming operators through form (1.1) or forms (1.1.1) - (.1.4), (2.1*) and analogues of forms (1.1.1) – (1.4) by type (2.1*) for more complex operators.

Here it is supposed to use a symbiosis of parallel actions and conventional calculations through sequential actions. This must be done through PrSCprt-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 PrSCprt -coding by a parallel computer system, through a Target-block with a PrSCprt -program operator -

$\begin{matrix} activation & activation & ... & activation \\ g_{21} & g_{22} & & g_{2n} \\ r_1 & r_2 & ... & r_n \end{matrix}$ PrSCprt(t) $\begin{matrix} u_1(t)w_1(t) & u_2(t)w_2(t) & ...u_n(t)w_n(t) \\ g_{11} & g_{21} & ... \; g_{n1} \\ activation & activation & ...activation \end{matrix}$, $r_j =$ $u_j(t)w_j(t)$, j = 1, 2, …, n, it will be possible to obtain the execution of the parallel actions $(u_1(t),u_2(t),...,u_n(t))$ with the desired target weights w(t)=($w_1(t)$, $w_2(t)_{,...,}w_n(t)$) with types g_{2j} of it. Each code for a neural network from a conventional computer we "bind" (match) to the corresponding value of current (or voltage). For PrSCprt-coding and PrSCprt-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 $\begin{matrix} activation & activation & ... & activation \\ g_{21} & g_{22} & & g_{2n} \\ r_1 & r_2 & ... & r_n \end{matrix}$ PrSCprt(t) $\begin{matrix} r_1 & r_2 & ... & r_n \\ g_{11} & g_{21} & ... & g_{n1} \\ activation & activation & ...activation \end{matrix}$, r_j = (UHF AC)$_j(t) := Q_j(t)$, we simultaneously enter the desired set of codes $Q_j(t)$, j = 1, 2, …, 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:

5) PrSCprt-program operators
6) PrtSCpr-program operators

Here are some of the PrSCprt-program operators:

1) Simultaneous assignment of the expressions $\tilde{p}$=($p_1|\mu_{\tilde{p}}(p_1)$, $p_2|\mu_{\tilde{p}}(p_2)_{, ...,}$ $p_n|\mu_{\tilde{p}}(p_n)$) to the variables $\tilde{x}$=($x_1|\mu_{\tilde{x}}(x_1)$, $x_2|\mu_{\tilde{x}}(x_2)_{, ...,}$ $x_n|\mu_{\tilde{x}}(x_n)$) with types g_j of it and *solution of task* Q_j *by* D_j with types g_{2j} of it simultaneously.

This is implemented via $\begin{matrix} task\ Q_1 & task\ Q_2 & & task\ Q_n \\ g_{21} & g_{22} & ... & g_{2n} \\ h_1 & h_2 & ... & h_n \end{matrix}$ PrSCprt

$x_1 :=\ x_2 :=\ \ldots x_n :=$
$g_{11}\quad g_{21}\quad \ldots\ g_{n1}$, h_j= *solution of task* Q_j *by* D_j with types g_{2j} of it, j
$p_1\quad p_2\quad \ldots\ p_n$

= 1, 2, …, n.

2) Simultaneous checking the set of conditions $\tilde{w}=(w_1|\mu_{\tilde{w}}(w_1), w_2|\mu_{\tilde{w}}(w_2), \ldots, w_n|\mu_{\tilde{w}}(w_n))$ for the set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2), \ldots, B_n|\mu_{\tilde{B}}(B_n))$ with types g_{j1} of it and *solution of task* Q_j *by* D_j with types g_{2j} of it simultaneously, j = 1, 2, …, n. Implemented via

$$\begin{array}{cccccccccc} task\ Q_1 & task\ Q_2 & & task\ Q_n & & IF\{B_1 w_1\}\ then & IF\{B_2 w_2\}\ then & \ldots IF\{B_n w_n\}\ then \\ g_{21} & g_{22} & \ldots & g_{2n} & \text{PrSCprt} & \mu_{11} & \mu_{21} & \ldots & \mu_{n1} \\ h_1 & h_2 & \ldots & h_n & & u_1 & u_2 & \ldots & u_n \end{array},$$

where u_i (i = 1, …, n) can be anything.

3) Similarly for loop operators and others.

PrSCprt-algorithm Examples:

3) Simultaneous addition and simultaneous parallel multiplication of sets elements (See point 1, 2 in **Math PrCself**

4) parallel pattern recognition: PrSCprt

$$\begin{array}{cccc} IF\{q_1 \in image\ archive_1\}\ then & IF\{q_2 \in image\ archive_2\}\ then & \ldots IF\{q_n \in image\ archive_n\}\ then \\ g_{11} & g_{21} & \ldots & g_{n1} \\ Name\ of\ q_1 & Name\ of\ q_2 & \ldots & Name\ of\ q_n \end{array}$$

The example of PrSCprt-program is

$$\begin{array}{cccccccccccccccc} & & x_1 := & x_2 := & \ldots x_n := & & & IF\{B_1 w_1\}\ then & IF\{B_2 w_2\}\ then & \ldots IF\{B_n w_n\}\ then & & & w_1 & w_2 & \ldots w_n \\ & \text{PrSCprt} & g_{11} & g_{21} & \ldots g_{n1} & \text{PrSCprt} & & g_{11} & g_{21} & \ldots\ g_{n1} & \ldots\text{PrSCprt} & g_{11} & g_{21} & \ldots g_{n1}, \\ \text{PrSCprt} & & p_1 & p_2 & \ldots\ p_n & & & u_1 & u_2 & \ldots\ u_n & & & w_1 & w_2 & \ldots w_n \\ & & & g_{11} & & & & & g_{21} & & & \ldots & & g_{n1} & \\ & & & r_1 & & & & & r_2 & & & \ldots & & r_n & \end{array}$$

PrS_3Cf- software operators will differ only just because aggregates $\{\tilde{w}\},\{\tilde{p}\},\{\tilde{B}\},\{\tilde{x}\}$ will be formed from corresponding PrSCprt-program operators in form (1.1) for more complex operators in forms (1.1.1) – (1.4) , (2.1*) and analogues of forms (1.1.1) - (1.4) by type (2.1*).

For example, $\begin{array}{cccccc} v_1\{R\} & v_2\{R\} & \ldots & v_n\{R\} & \{R\}_1 & \{R\}_2 \\ g_{11} & g_{21} & & g_{n1} & g_{11} & g_{21} \\ \{R\}_1 & \{R\}_2 & \ldots & \{R\}_n & v_1\{R\} & v_2\{R\} \end{array}$ PrSprt $\begin{array}{ccc} \ldots & \{R\}_n \\ \ldots & g_{n1} \\ \ldots & v_n\{R\} \end{array}$ is

the capacity in itself of the second type if $v_j\{R\}$ is a program capable of generating $\{R\}_j$.

The example of self-program of the first type is

$$\begin{array}{rrrl} x_1 := & x_2 := & \dots x_n := & \\ \text{PrSCprt } g_{11} & g_{21} & \dots\ g_{n1} & \\ p_1 & p_2 & \dots\ p_n & \end{array} \quad \begin{array}{rrrr} IF\{B_1 w_1\}\ then & IF\{B_2 w_2\}\ then & \dots IF\{B_n w_n\}\ then & \\ \text{PrSCprt } g_{11} & g_{21} & \dots\ g_{n1} & \\ u_1 & u_2 & \dots\ u_n & \end{array} \quad \begin{array}{rrrl} w_1 & w_2 & \dots w_n & \\ \dots \text{PrSCprt} g_{11} & g_{21} & \dots g_{n1}, & \\ w_1 & w_2 & \dots w_n & \end{array}$$

$$\text{PrSCprt} \quad g_{11} \qquad\qquad g_{21} \qquad\qquad \dots \quad g_{n1} \quad .$$

$$\begin{array}{rrrl} x_1 := & x_2 := & \dots x_n := & \\ \text{PrSCprt } g_{11} & g_{21} & \dots\ g_{n1} & \\ p_1 & p_2 & \dots\ p_n & \end{array} \quad \begin{array}{rrrr} IF\{B_1 w_1\}\ then & IF\{B_2 w_2\}\ then & \dots IF\{B_n w_n\}\ then & \\ \text{PrSCprt } g_{11} & g_{21} & \dots\ g_{n1} & \\ u_1 & u_2 & \dots\ u_n & \end{array} \quad \begin{array}{rrrl} w_1 & w_2 & \dots w_n & \\ \dots \text{PrSCprt} g_{11} & g_{21} & \dots g_{n1} & \\ w_1 & w_2 & \dots w_n & \end{array}$$

PrSCprt-coding: 1) set A_i to set B_i with types g_{ij} of it, 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, …, n; j = 1, 2, …, m).

For example, $\begin{array}{rrrl} B_1 & B_2 & \dots B_n & \\ g_{12} & g_{22} & \dots g_{n2} & \text{PrSCprt} \\ A_1 & A_2 & \dots A_n & \end{array} \begin{array}{rrrl} A_1 & A_2 & \dots A_n & \\ g_{11} & g_{21} & \dots g_{n1} & . \\ B_1 & B_2 & \dots B_n & \end{array}$

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

PrCSelf-coding: 1) set A_i to set A_i with types g_{ij} of it, 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, $\begin{array}{rrrl} A_1 & A_2 & \dots A_n & \\ g_{12} & g_{22} & \dots g_{n2} & \text{PrSCprt} \\ A_1 & A_2 & \dots A_n & \end{array} \begin{array}{rrrl} A_1 & A_2 & \dots A_n & \\ g_{11} & g_{21} & \dots g_{n1} & . \\ A_1 & A_2 & \dots A_n & \end{array}$

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 PrSCprt-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 PrSCprt -Networks and series-parallel. Codes from a conventional type computer system will be used via PrSCprt -connectors in PrSCprt - coding, for example:

$$\begin{array}{rrrrl} activation & activation & \dots & activation & \\ g_{21} & g_{22} & & g_{2n} & \text{PrSCprt(t)} \\ r_1 & r_2 & \dots & r_n & \end{array} \begin{array}{rrrl} r_1 & r_2 & \dots & r_n \\ g_{11} & g_{21} & \dots & g_{n1} \\ activation & activation & \dots & activation \end{array}, r_j =$$

$(\text{UHF AC})_j(t) := Q_j(t)$, UHF AC field activation is used.

Consider the dynamic PrSCprt and $PrS_3Cf(t)$ 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 $\widetilde{x(t)}=(x_1(t)|\mu_{\widetilde{x(t)}}(x_1(t)),$

$x_2(t)|\mu_{\tilde{x}(t)}(x_2(t))$, …, $x_n(t)|\mu_{\tilde{x(t)}}(x_n(t))$) with types g_{i1} of it and

solution of task Q_j *by* D_j with types g_{2j} of it simultaneously, j = 1, 2, …, n, is

$$\text{implemented through } \begin{matrix} task\ Q_1 & task\ Q_2 & & task\ Q_n \\ g_{21} & g_{22} & \cdots & g_{2n} \\ h_1 & h_2 & \ldots & h_n \end{matrix} \text{ PrSCprt}$$

$$\begin{matrix} x_1(t) := & x_2(t) := & \ldots x_n(t) := \\ g_{11} & g_{21} & \cdots \ g_{n1} \\ p_1(t) & p_2(t) & \ldots \ p_n(t) \end{matrix} .$$

2. The process of simultaneous check the set of conditions $\tilde{w}$=($w_1|\mu_{\tilde{w}}(w_1)$, $w_2|\mu_{\tilde{w}}(w_2)$, …, $w_n|\mu_{\tilde{w}}(w_n)$) for the set of expressions $\widetilde{B(t)}$=($B_1(t)|\mu_{\widetilde{B(t)}}(B_1(t))$, $B_2(t)|\mu_{\tilde{B}(t)}(B_2(t))$, …, $B_n(t)|\mu_{\widetilde{B(t)}}(B_n(t))$) and *solution of task* Q_j *by* D_j with types g_{2j} of it simultaneously, j = 1, 2, …, n, is implemented through $\begin{matrix} task\ Q_1 & task\ Q_2 & & task\ Q_n \\ g_{21} & g_{22} & \cdots & g_{2n} \\ h_1 & h_2 & \ldots & h_n \end{matrix}$

$$\text{PrSCprt } \begin{matrix} IF\{B_1(t)w_1(t)\}\ then & IF\{B_2(t)w_2(t)\}\ then & \ldots IF\{B_n(t)w_n(t)\}\ then \\ g_{11} & g_{21} & \cdots \quad g_{n1} \\ u_1(t) & u_2(t) & \ldots \quad u_n(t) \end{matrix} \text{, where u(t) =}$$

$(u_1(t),u_2(t),\ldots,u_n(t))$ can be any.

3.Similarly for loop operators and others.

p-analogue of $PrS_3Cf(t)$– software operators will differ only in that the aggregates $\{\widetilde{w(t)}\},\{\widetilde{p(t)}\},\{\widetilde{B(t)}\},\{\widetilde{x(t)}\}$ will be formed from corresponding processes PrSCprt(t) for the above-mentioned programming operators through p-analogue of form (1.1) or p-analogue of forms (1.1.1) – (1.4) for more complex operators, (2.1*) and analogues of forms (1.1.1) - (1.4) by type (2.1*).

Consider PrfftSpr-program operators. The ideology of PrtSCpr and Prt_{S_4Cf} is parallel analogue of t_{S_4Cf} [14] can be used for programming. Here are some of the PrtSCpr -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 $\tilde{x}$=($x_1|\mu_{\tilde{x}}(x_1)$, $x_2|\mu_{\tilde{x}}(x_2)$, …, $x_n|\mu_{\tilde{x}}(x_n)$). It's implemented through $\begin{matrix} x_1 & x_2 & \ldots & x_n \\ g_{11} & g_{21} & \cdots & g_{n1} \\ := p_1 & := p_2 & \ldots & := p_n \end{matrix}$ PrSCprt.

2) Simultaneous expelling checks the set of conditions $\tilde{w}=(w_1|\mu_{\tilde{w}}(w_1), w_2|\mu_{\tilde{w}}(w_2), \ldots, w_n|\mu_{\tilde{w}}(w_n))$ for the set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2), \ldots, B_n|\mu_{\tilde{B}}(B_n))$. It's implemented through

$$\begin{matrix} d_1 & d_2 & \ldots & d_n \\ g_{11} & g_{21} & \ldots & g_{n1} \end{matrix} \text{ PrSCprt,}$$
$$IF\{B_1 w_1\}\ then\ u_1 \quad IF\{B_2 w_2\}\ then\ u_2 \quad \ldots IF\{B_n w_n\}\ then\ u_n$$

where u_i (i = 1, ..., n) can be anything.

3) Similarly for loop operators and others.

Prt_{S_4Cf} – software operators will differ only just because aggregates $\{\tilde{w}\},\{\tilde{p}\},\{\tilde{B}\},\{\tilde{x}\}$ will be formed from corresponding PrtSCpr program operators in form (1.1) for more complex operators in forms (1.1.1) – (1.4) , (2.1*) and analogues of forms (1.1.1) - (1.4) by type (2.1*).

Consider hierarchical PrtSpr-program operator

$$\begin{matrix} C_1(t) & C_2(t) & \ldots C_m(t) \\ g_{12} & g_{22} & \ldots\ g_{m2} \\ D_1(t) & D_2(t) & \ldots D_m(t) \end{matrix} \text{ PrSCrt}_2\text{(t)} =$$

$$\left\{ \begin{matrix} \sum_{i=1}^{m} Q_i(t) + & \{\} & \{\} & \ldots & \{\} \\ & g_{12} & g_{22} & \ldots & g_{m2} \\ & D_1(t) - D_1(t) \cap C_1(t) & D_2(t) - D_2(t) \cap C_2(t) & \ldots & D_m(t) - D_2(t) \cap C_m(t) \\ & \sum_{i=1}^{m} (C_i(t) - D_i(t) \cap C_i(t)) - (D_i(t) - D_i(t) \cap C_i(t)) & & & \end{matrix} \right\} \text{PrSCrt(t)}$$

, where $Q_i(t)$ is oself-set for $(D_i(t) \cap C_i(t))$ [14].

Consider the PrtSCpr(t) and $Prt(t)_{S_4Cf}$ programming at time t.

1. The process of simultaneous assignment of the expressions $\widetilde{p(t)}=(p_1(t)|\mu_{\widetilde{p(t)}}(p_1(t)), p_2(t)|\mu_{\widetilde{p(t)}}(p_2(t)), \ldots, p_n(t)|\mu_{\widetilde{p(t)}}(p_n(t)))$ to the variables $\widetilde{x(t)}=(x_1(t)|\mu_{\widetilde{x(t)}}(x_1(t)), x_2(t)|\mu_{\tilde{x}(t)}(x_2(t)), \ldots, x_n(t)|\mu_{\widetilde{x(t)}}(x_n(t)))$ is implemented through

$$\begin{matrix} x_1(t) & x_2(t) & \ldots & x_n(t) \\ \mu_{11} & \mu_{21} & \ldots & \mu_{n1} \\ := p_1(t) & := p_2(t) & \ldots := & p_n(t) \end{matrix} \text{ PrSCprt(t).}$$

2. The process of simultaneous check the set of conditions $\widetilde{u(t)}=(u_1(t)|\mu_{\widetilde{u(t)}}(u_1(t)), u_2(t)|\mu_{\tilde{u}(t)}(u_2(t)), \ldots, u_n(t)|\mu_{\widetilde{u(t)}}(u_n(t)))$ for the set of expressions $\widetilde{B(t)}=(B_1(t)|\mu_{\widetilde{B(t)}}(B_1(t)), B_2(t)|\mu_{\tilde{B}(t)}(B_2(t)), \ldots, B_n(t)|\mu_{\widetilde{B(t)}}(B_n(t)))$ is implemented through

$$\begin{matrix} d_1(t) & d_2(t) & \dots & d_n(t) & \\ g_{11} & g_{21} & \dots & g_{n1} & \text{PrSCprt(t),} \\ IF\{B_1(t)u_1(t)\}\ then\ w_1(t) & IF\{B_2(t)u_2(t)\}\ then\ w_2(t) & \dots IF\{B_n(t)u_n(t)\}\ then\ w_n(t) & & \end{matrix}$$

where w_i(t) (i = 1, ..., n) can be anything.

3.Similarly for loop operators and others.

$Prt(t)_{S_4Cf}$ – software operators will differ only in that the aggregates $\widetilde{x(t)}$,$\widetilde{p(t)}$, $\widetilde{B(t)}$,$\widetilde{w(t)}$ will be formed from corresponding processes PrSCprt(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 analogues of forms (1.1.1) - (1.4) by type (2.1*).

Consider PrSCrt(t)- program operators

$$\{ \begin{matrix} q\begin{pmatrix} u & u & u & u \\ g_1 & g_2 PrSCpr g_1 & g_2 \\ u & u & u & u \end{pmatrix} \\ W_q \end{matrix} fSprt_q^{E_q} \begin{pmatrix} u & u & u & u \\ g_1 & g_2 PrSCpr g_1 & g_2 \\ u & u & u & u \end{pmatrix}', \ fSprt_{d_r}^{\{E^{ex}{}_l{}^{d_r}\}}\left(E_{in}{}^{d_r}\right) \}$$

$fSprt_{t_0}$. —program structure example, where the assemblage point d_r is the cursor, it is quite complex Cself—program.

Remark. Energy with type of containment g_1, g_2 of a living organism other than humans:

$$\text{PrC(r, u}(E_q), g_1, g_2) = \{ \begin{matrix} q\begin{pmatrix} u & u & u & u \\ g_1 & g_2 PrSCpr g_1 & g_2 \\ u & u & u & u \end{pmatrix} \\ W_q \end{matrix} Sprt_q^{E_q} \begin{pmatrix} u & u & u & u \\ g_1 & g_2 PrSCpr g_1 & g_2 \\ u & u & u & u \end{pmatrix}', \ fSprt_{d_r}^{\{E^{ex}{}_l{}^{d_r}\}}\left(E_{in}{}^{d_r}\right) \} fSprt_{t_0}$$

(**$_{F.3}$).

Energy with type of containment μ_1, μ_2 of a living organism of a person:

fPrhff(r, u(E_q), μ_1, μ_2) =

$$\{ \begin{matrix} q\begin{pmatrix} u & u & u & u \\ g_1 & g_2 PrSCpr g_1 & g_2 \\ u & u & u & u \end{pmatrix} \\ W_q \end{matrix} Sprt_q^{E_q} \begin{pmatrix} u & u & u & u \\ g_1 & g_2 PrSCpr g_1 & g_2 \\ u & u & u & u \end{pmatrix}', \ fSprt_{d_{r\,(self\left(E_{in}{}^{d_r}\right))}}^{\{E^{ex}{}_l{}^{d_r}\}} \}$$

$fSprt_{t_0}$ (***$_{F.3}$).

$\begin{matrix} u & u \\ g_1 & g_2 \\ u & u \end{matrix} PrSCpr \begin{matrix} u & u \\ g_1 & g_2 \\ u & u \end{matrix}$ -internal energy with type of containment μ_1, μ_2 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.

($**_{F.3}$), ($***_{F.3}$) can be interpreted as PrSCprt- program operators.

PrffSCprt and $Prff S_3 Cf$ programming

The ideology of PrffSCprt and $Prff S_3 Cf$ can be used for programming. Here are some of the PrffSCprt programming operators.

1. Simultaneous fuzzy assignment of the expressions $\tilde{p}$=($p_1|\mu_{\tilde{p}}(p_1)$, $p_2|\mu_{\tilde{p}}(p_2)$, …, $p_n|\mu_{\tilde{p}}(p_n)$) to the variables $\tilde{x}$=($x_1|\mu_{\tilde{x}}(x_1)$, $x_2|\mu_{\tilde{x}}(x_2)$, …, $x_n|\mu_{\tilde{x}}(x_n)$) with types of assignment v_{j1}, measure of fuzziness μ_{j1} and expelling of the expressions $\tilde{r}$ =($r_1|\mu_{\tilde{r}}(r_1)$, $r_2|\mu_{\tilde{r}}(r_2)|_{,\ldots,}$ $r_n|\mu_{\tilde{r}}(r_n)$) from the variables $\tilde{y}$=($y_1|\mu_{\tilde{y}}(y_1)$, $y_2|\mu_{\tilde{y}}(y_2)|_{,\ldots,}y_n|\mu_{\tilde{y}}(y_n)$) with types of expelling v_{2j}& measure of fuzziness μ_{j1} simultaneously, j = 1, 2, …, n. simultaneously. This is implemented via
$\begin{matrix} y_1 & y_2 & \ldots & y_n \\ v_{11} & v_{21} & \ldots & v_{n1} \\ \mu_{11} & \mu_{21} & \ldots & \mu_{n1} \\ := r_1 & := r_2 & \ldots := & r_n \end{matrix}$ PrffSCprt $\begin{matrix} x_1 := & x_2 := & \ldots & x_n := \\ v_{11} & v_{21} & \ldots & v_{n1} \\ \mu_{11} & \mu_{21} & \ldots & \mu_{n1} \\ p_1 & p_2 & \ldots & p_n \end{matrix}$.
2. Simultaneous fuzzy checking the 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 set of expressions $\tilde{B}$=($B_1|\mu_{\tilde{B}}(B_1)$, $B_2|\mu_{\tilde{B}}(B_2)$, …, $B_n|\mu_{\tilde{B}}(B_n)$)

and *solution of task* Q_j *by* D_j simultaneously. Implemented via

task Q_1	*task* Q_2		*task* Q_n		$IF\{B_1g_1\}$ *then*	$IF\{B_2g_2\}$ *then*		$IF\{B_ng_n\}$ *then*
v_{11}	v_{11}	...	v_{11}	PrffSCprt	v_{11}	v_{21}	...	v_{n1}
μ_{21}	μ_{22}		μ_{2n}		μ_{11}	μ_{21}	...	μ_{n1}
h_1	h_2	...	h_n		w_1	w_2	...	w_n

,

where w_i (i = 1, …, n) can be anything, h_j= *solution of task* Q_j *by* D_j, j = 1, 2, …, n.

3. Similarly for loop operators and others.

p-analogue of $PrffS_3Cf-$ software operators will differ only in that the aggregates $\{\tilde{g}\},\{\tilde{p}\},\{\tilde{B}\},\{\tilde{x}\}$ will be formed from the corresponding PrffSCprt-program operators in p-analogue of form (1.1) and for more complex operators in the p-analogue of forms (1.1.1) –(1.4), (2.1*) and analogues 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.

About fuzzy dynamic PrffSCprt and PrffS₃Cf(t) programming.

The ideology of fuzzy dynamic PrffSCprt and PrffS$_3$Cf(t) can be used for programming:

1. The process of simultaneous assignment of the expressions $\widetilde{p(t)}$=($p_1(t)|\mu_{\widetilde{p(t)}}(p_1(t))$, $p_2(t)|\mu_{\widetilde{p(t)}}(p_2(t))$, …, $p_n(t)|\mu_{\widetilde{p(t)}}(p_n(t))$) to the variables $\widetilde{x(t)}$=($x_1(t)|\mu_{\widetilde{x(t)}}(x_1(t))$, $x_2(t)|\mu_{\tilde{x}(t)}(x_2(t))$, …, $x_n(t)|\mu_{\widetilde{x(t)}}(x_n(t))$) and *solution of task* Q_j *by* D_j simultaneously is implemented through

task Q_1	*task* Q_2		*task* Q_n		$x_1(t) :=$	$x_2(t) :=$		$x_n(t) :=$
v_{11}	v_{11}	...	v_{11}	PrffSCprt(t)	v_{11}	v_{21}	...	v_{n1}
μ_{21}	μ_{22}		μ_{2n}		μ_{11}	μ_{21}	...	μ_{n1}
h_1	h_2	...	h_n		$p_1(t)$	$p_2(t)$	...	$p_n(t)$

, h_j=

solution of task Q_j *by* D_j, j = 1, 2, …, n.

2. The process of simultaneous check the set of conditions $\widetilde{u(t)}$=($u_1(t)|\mu_{\widetilde{u(t)}}(u_1(t))$, $u_2(t)|\mu_{\widetilde{u(t)}}(u_2(t))$, …, $u_n(t)|\mu_{\widetilde{u(t)}}(u_n(t))$) for a set of expressions $\widetilde{B(t)}$=($B_1(t)|\mu_{\widetilde{B(t)}}(B_1(t))$,

$B_2(t)|\mu_{\tilde{B}(t)}(B_2(t))$, ..., $B_n(t)|\mu_{\widetilde{B(t)}}(B_n(t))$) and *solution of task* Q_j *by* D_j simultaneously is implemented through

task Q_1	*task* Q_2		*task* Q_n		$IF\{B_1(t)u_1(t)\}$ *then*	$IF\{B_2(t)u_2(t)\}$ *then*		$IF\{B_n(t)u_n(t)\}$ *then*
v_{11}	v_{11}	...	v_{11}	PrffSCprt(t)	v_{11}	v_{21}	...	v_{n1}
μ_{21}	μ_{22}		μ_{2n}		μ_{11}	μ_{21}	...	μ_{n1}
h_1	h_2	...	h_n		$w_1(t)$	$w_2(t)$	...	$w_n(t)$

here w(t) = $(w_1(t),w_2(t),...,w_n(t))$ can be any, h_j= *solution of task* Q_j *by* D_j, j = 1, 2, …, n.

3.Similarly for loop operators and others.

p – analogue of $Prff S_3 Cf(t)$– software operators will differ only in that the aggregates $\{u(\widetilde{t})\}, \{p(\widetilde{t})\}, \{B(\widetilde{t})\}, \{x(\widetilde{t})\}$ will be formed from corresponding processes PrffSCprt(t) for the above-mentioned programming operators through p-analogue of form (1.1) or p-analogue of forms (1.1.1) - (.1.4), (2.1*) and analogues of forms (1.1.1) – (1.4) by type (2.1*) for more complex operators.

The usage of PrffSCprt-elements for networks.

A. Galushkin's comprehensive monograph [21] 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. PrffSCprt-networks (SmnPrffSCprt) are a PrffSCprt-structure that can be built for the required weights. PrffSCprt-OS (PrffSCprt operating system) uses PrffSCprt-coding and PrffSCprt-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. PrffSCprt-coding (or PrffCSelf-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. PrffSCprt-translation is carried out by inversion. In

this case, PrffCSelf-coding and PrffCSelf-translation will be more stable. The target weights $g_i(t)$ in

$$\begin{matrix} r_1 & r_2 & & r_n \\ v_{11} & v_{21} & \dots & v_{n1} \\ \mu_{11} & \mu_{12} & & \mu_{1n} \\ u & u & \dots & u \end{matrix} \overleftarrow{PrffSCprt(t)} \begin{matrix} u & u & & u \\ v_{11} & v_{21} & \dots & v_{n1} \\ \mu_{21} & \mu_{22} & & \mu_{2n} \\ r_1 & r_2 & \dots & r_n \end{matrix},$$

$r_{i=}$ *activation with* $g_i(t)$, u = SmnPrffSCprt, are chosen for necessary tasks. We will not touch on the issues of applications, or network optimization. They are described in detail by Galushkin [21]. 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 PrffSCprt (designation - mnPrffSCprt) for simple information processing. More complex executing programs are used for mnPrffSCprt nodes. PrffSCprt-threshold element –sgn(

$$\text{PrffSCprt(t)} \begin{matrix} g_1(t)w_1(t) & g_2(t)w_2(t) & \dots & g_n(t)w_n(t) \\ v_{11} & v_{21} & \dots & v_{n1} \\ \mu_{11} & \mu_{21} & \dots & \mu_{n1} \\ x_1 & x_2 & \dots & x_n \end{matrix}$$

), x=($x_1, x_2, \dots, x_n$) - mnPrffSCprt,

$\widetilde{W(t)}$=($w_1(t)|\mu_{\widetilde{w(t)}}(w_1(t))$, $w_2(t)|\mu_{\widetilde{w(t)}}(w_2(t))$, …, $w_n(t)|\mu_{\widetilde{w(t)}}(w_n(t))$).

– source signals values, $\{g(t)\} = (g_1(t), g_2(t), \dots, g_n(t))$ – PrffSCprt-synapses weights. The first level of mnPrffSCprt consists of simple mnPrffSCprt. The second level of mnPrffSCprt consists of q = $\begin{matrix} D_1 & D_2 & \dots & D_n \\ v_{11} & v_{21} & \dots & v_{n1} \\ \mu_{11} & \mu_{21} & \dots & \mu_{n1} \\ v & v & \dots & v \end{matrix} \overleftarrow{PrffSCprt(t)}$

$\begin{matrix} v & v & & v \\ v_{11} & v_{21} & \dots & v_{n1} \\ \mu_{11} & \mu_{21} & \dots & \mu_{n1} \\ D_1 & D_2 & \dots & D_n \end{matrix}$, v = mnPrffSCprt, – PrffSCprt-node of mnPrffSCprt in range D = ($D_1, D_2, \dots, D_n$), D- fcapacity for mnPrffSCprt node. The third level of mnPrffSCprt consists of $\begin{matrix} D_1 & D_2 & \dots & D_n \\ v_{11} & v_{21} & \dots & v_{n1} \\ \mu_{11} & \mu_{21} & \dots & \mu_{n1} \\ q & q & \dots & q \end{matrix} \overleftarrow{PrffSCprt(t)} \begin{matrix} q & q & & q \\ v_{11} & v_{21} & \dots & v_{n1} \\ \mu_{11} & \mu_{21} & \dots & \mu_{n1} \\ D_1 & D_2 & \dots & D_n \end{matrix}$

-PrffSCprt^2- node of mnPrffSCprt in range D, thus D becomes fcapacity of itself in itself as an element for mnPrffSCprt. For our networks, it is sufficient to use PrffSCprt^2- nodes of mnPrffSCprt, but self-level is higher in living organisms,

particularly PrffSCprt^{n}-, n≥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 F.3.5.0. A neural network can be thought of as a learnable parallel fuzzy dynamic operator.

Remark F.3.5.1. 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.

mnPrffSCprt contains $\begin{matrix} \text{ffceprogram}_1(t) & \text{ffceprogram}_2(t) & & \text{ffceprogram}_n(t) \\ v_{11} & v_{21} & \cdots & v_{n1} \\ \mu_{11} & \mu_{21} & \cdots & \mu_{n1} \\ \text{mnPrffSCprt} & \text{mnPrffSCprt} & \ldots & \text{mnPrffSCprt} \end{matrix}$ PrffSCprt(t)

$\begin{matrix} \text{mnPrffSCprt} & \text{mnPrffSCprt} & & \text{mnPrffSCprt} \\ v_{11} & v_{21} & \cdots & v_{n1} \\ \mu_{11} & \mu_{21} & \cdots & \mu_{n1} \\ \text{ffceprogram}_1(t) & \text{ffceprogram}_2(t) & \ldots & \text{ffceprogram}_n(t) \end{matrix}$ ffceprogram –executing program in PrffSCprt-OS. PrffSCprt-OS (or PrffCSelf-OS) is based on PrffSCprt-assembly language (or Prself-assembly language), which is based on assembly language through PrffSCprt-approach in turn, if the base of elements of PrffSCprt-networks is sufficient. The ffceprograms are in PrffSCprt-programming environments (or PrffCSelf-programming environments), but this question and PrffSCprt-networks base will be considered in the following monographs. In particular, ffceprograms may contain PrffSCprt- programming operators. In mnPrffSCprt cores, the constant memory PrffSCprt with correspondent ffceprograms depending on mnPrffSCprt.

The OS (operating system) and the principles and modes of operation of the PrffSCprt-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 PrffSCprt – physics and special neural networks with artificial neurons operating in normal and PrffSCprt-modes. Let's denote this model through SmnPrffSCprt. To

do this, it’s proposed to use mnPrffSCprt of different levels; for example, for the usual mode, mnPrffSCprt 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 PrffSCprt–mode with the desired "target weight" is realized in this section, etc., to the center. In the case of a monster during the test, SmnPrffSCprt is activated with the desired "target weight." Here are realized other tasks also. To reach the fself-energy level, the mode $^{\text{SmnPrffSCprt}}_{\text{SmnPrffSCprt}}Sprt^{\text{SmnPrffSCprt}}_{\text{SmnPrffSCprt}}$ is used. In normal mode, it’s planned to carry out the movement of SmnPrffSCprt 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 SmnPrffSCprt 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 SmnPrffSCprt. SmnPrffSCprt is represented by a neural network that extends from the center of one of the main clusters of PrffSCprt - 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 SmnPrffSCprt's actions is below the operator's cab. In PrffSCprt – mode, the entire network or its sections are PrffSCprt – activated to perform specific tasks, in particular, with "target weights." In the target, block used PrffSCprt -coding, PrffSCprt -translation for activation of all networks to "target weights" simultaneously, then –the reset of this PrffSCprt-coding after activation. Unfortunately, triodes are not suitable for PrffSCprt -neural networks. In the most primitive case, usual separators with corresponding resistances and cores for ffeprograms may be used instead triodes since there is no necessity to unbend the alternating current to direct. The PrffSCprt-operative memory belt is disposed around a central core of SmnPrffSCprt. There are PrffSCprt-coding, PrffSCprt-translation, and PrffSCprt-realize of eprograms and the programs from the archives without extraction, PrffSCprt-

coding and PrffSCprt-translation may be used in high-intensity, ultra-short optical pulses laser of Nobel laureates 2018-year Gerard Mourou, Donna, Strickland. PrffSCprt – structure or an eprogram if one is present of needed «target weight» are taken in target block at PrffSCprt – activation of the networks.

$\underset{\text{SmnPrffSCprt},f}{\overset{activation}{Sprt}}{}^{\text{SmnPrffSCprt},f}_{activation}$ derives SmnPrffSCprt to the self-level boundary with target weight f.

It's used an alternating current of above high frequency , which can work with PrffSCprt – structures in PrffSCprt–modes by its nature to activate the networks or some of its parts in PrffSCprt–modes and locally using PrffSCprt–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 PrffSCprt-networks is realized to indicate "target weights."

Program operators PrffSCprt, PrfftSCpr.

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

Using conventional PrffSCprt -coding by a parallel computer system, through a Target-block with a PrffSCprt -program operator -

$$\begin{matrix} activation & activation & & activation \\ v_{11} & v_{21} & \cdots & v_{n1} \\ \mu_{11} & \mu_{21} & \cdots & \mu_{n1} \\ g_1(t)w_1(t) & g_2(t)w_2(t) & \cdots & g_n(t)w_n(t) \end{matrix} \text{PrffSCprt(t)} \begin{matrix} g_1(t)w_1(t) & g_2(t)w_2(t) & \cdots & g_n(t)w_n(t) \\ v_{11} & v_{21} & \cdots & v_{n1} \\ \mu_{11} & \mu_{21} & \cdots & \mu_{n1} \\ activation & activation & \cdots & activation \end{matrix}$$

, it will be possible to obtain the execution of the parallel fuzzy actions $(g_1(t), g_2(t), \ldots, g_n(t))$ with the desired target weights w(t)=($w_1(t), w_2(t), \ldots, w_n(t)$). Each code for a neural network from a conventional computer we "bind" (match) to the corresponding value of current (or voltage). For PrffSCprt-coding and PrffSCprt-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

$$\begin{matrix} activation & activation & & activation \\ v_{11} & v_{21} & \cdots & v_{n1} \\ \mu_{11} & \mu_{21} & \cdots & \mu_{n1} \\ u_1 & u_2 & \cdots & u_n \end{matrix} \text{ PrffSCprt(t) } \begin{matrix} u_1 & u_2 & & u_n \\ v_{11} & v_{21} & \cdots & v_{n1} \\ \mu_{11} & \mu_{21} & \cdots & \mu_{n1} \\ activation & activation & & ...activation \end{matrix} , \mathrm{u_j} =$$

$(\text{UHF AC})_j(t) := Q_j(t)$ we simultaneously enter the desired fuzzy set of codes $Q_j(t)$, j = 1, 2, …, 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) PrffSCprt-program operators
2) PrfftSCpr-program operators

Here are some of the PrffSCprt-program operators:

1) Simultaneous fuzzy p-assignment of the fuzzy expressions $\tilde{p}=(\mathrm{p_1}|\mu_{\tilde{p}}(\mathrm{p_1}), \mathrm{p_2}|\mu_{\tilde{p}}(\mathrm{p_2}), \ldots, \mathrm{p_n}|\mu_{\tilde{p}}(\mathrm{p_n}))$ to the variables $\tilde{x}=(\mathrm{x_1}|\mu_{\tilde{x}}(\mathrm{x_1}), \mathrm{x_2}|\mu_{\tilde{x}}(\mathrm{x_2}), \ldots, \mathrm{x_n}|\mu_{\tilde{x}}(\mathrm{x_n}))$. This is implemented via

$$\begin{matrix} x_1 & x_2 & & x_n \\ v_{11} & v_{21} & \cdots & v_{n1} \\ \mu_{11} & \mu_{21} & \cdots & \mu_{n1} \\ := p_1 & := p_2 & \ldots := & p_n \end{matrix} \text{ PrffSCprt } \begin{matrix} x_1 := & x_2 := & & x_n := \\ v_{11} & v_{21} & \cdots & v_{n1} \\ \mu_{11} & \mu_{21} & \cdots & \mu_{n1} \\ p_1 & p_2 & \cdots & p_n \end{matrix} .$$

2) Simultaneous fuzzy checking the fuzzy set of conditions $\tilde{g}=(\mathrm{g_1}|\mu_{\tilde{g}}(\mathrm{g_1}), \mathrm{g_2}|\mu_{\tilde{g}}(\mathrm{g_2}), \ldots, \mathrm{g_n}|\mu_{\tilde{g}}(\mathrm{g_n}))$ for the fuzzy set of expressions $\tilde{B}=(\mathrm{B_1}|\mu_{\tilde{B}}(\mathrm{B_1}), \mathrm{B_2}|\mu_{\tilde{B}}(\mathrm{B_2}), \ldots, \mathrm{B_n}|\mu_{\tilde{B}}(\mathrm{B_n}))$ and *solution of task Q_j by D_j* simultaneously.

Implemented via $\begin{matrix} task\ Q_1 & task\ Q_2 & & task\ Q_n \\ \mu_{21} & \mu_{22} & \cdots & \mu_{2n} \\ h_1 & h_2 & \ldots & h_n \end{matrix}$ PrffSCprt

$$\begin{matrix} IF\{B_1 g_1\}\ then & IF\{B_2 g_2\}\ then & & IF\{B_n g_n\}\ then \\ v_{11} & v_{21} & \cdots & v_{n1} \\ \mu_{11} & \mu_{21} & \cdots & \mu_{n1} \\ w_1 & w_2 & \cdots & w_n \end{matrix} ,$$ where w_i (i = 1, …, n) can be

anything, h_j= *solution of task* Q_j *by* D_j with measure of fuzziness μ_{2j}, j = 1, 2, …, n.

3) Similarly for loop operators and others.

PrffSCprt-algorithm Examples:

1) Simultaneous addition and simultaneous parallel fuzzy multiplication of fuzzy sets elements (See point 1, 2 in **Math PrffCSelf**
2) parallel fuzzy pattern recognition: PrffSCprt

$$\begin{array}{ccccc} IF\left\{q_1 \in image\ archive_1\right\}\ then & IF\left\{q_2 \in image\ archive_2\right\}\ then & \ldots & IF\left\{q_n \in image\ archive_n\right\}\ then \\ v_{11} & v_{21} & & v_{n1} \\ \mu_{11} & \mu_{21} & \ldots & \mu_{n1} \\ Name\ of\ q_1 & Name\ of\ q_2 & \ldots & Name\ of\ q_n \end{array}$$

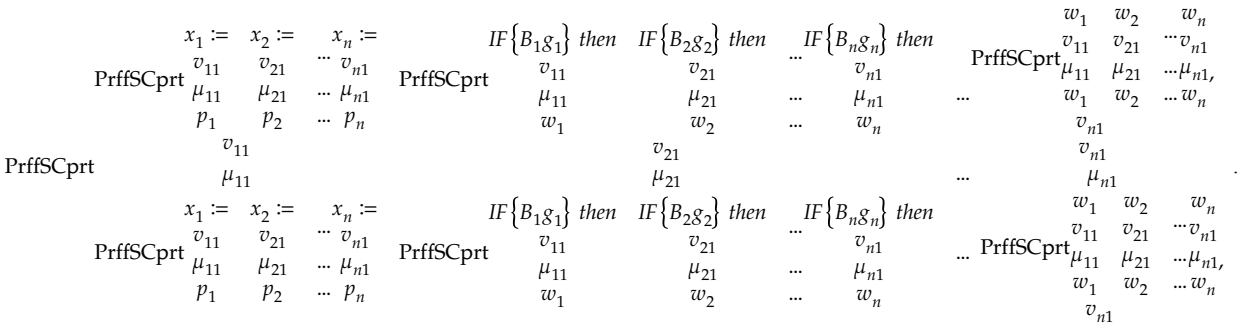
The example of PrffSCprt-program is

$$\begin{array}{l} x_1 := \\ v_{11} \\ \mu_{11} \\ p_1 \\ v_{11} \\ \mu_{11} \\ r_1 \end{array} \begin{array}{l} x_2 := \\ v_{21} \\ \mu_{21} \\ p_2 \end{array} \begin{array}{l} \ldots\ x_n := \\ \ldots\ v_{n1} \\ \ldots\ \mu_{n1} \\ \ldots\ p_n \end{array} \begin{array}{l} \text{PrffSCprt} \\ \text{PrffSCprt} \end{array} \begin{array}{l} IF\left\{B_1 g_1\right\}\ then \\ v_{11} \\ \mu_{11} \\ w_1 \\ v_{21} \\ \mu_{21} \\ r_2 \end{array} \begin{array}{l} IF\left\{B_2 g_2\right\}\ then \\ v_{21} \\ \mu_{21} \\ w_2 \end{array} \begin{array}{l} \ldots\ IF\left\{B_n g_n\right\}\ then \\ \ldots\ v_{n1} \\ \ldots\ \mu_{n1} \\ \ldots\ w_n \end{array} \begin{array}{l} \ldots \\ \text{PrffSCprt} \\ \ldots \end{array} \begin{array}{l} w_1 \\ v_{11} \\ \mu_{11} \\ w_1 \\ v_{n1} \\ \mu_{n1} \\ r_n \end{array} \begin{array}{l} w_2 \\ v_{21} \\ \mu_{21} \\ w_2 \end{array} \begin{array}{l} \ldots\ w_n \\ \ldots v_{n1} \\ \ldots \mu_{n1}, \\ \ldots w_n \end{array}$$

$PrffS_3Cf-$ software operators will differ only just because fuzzy aggregates $\{\tilde{g}\},\{\tilde{p}\},\{\tilde{B}\},\{\tilde{x}\}$ will be formed from corresponding PrffSCprt-program operators in form (1.1) for more complex operators in forms (1.1.1) – (1.4) , (2.1*) and analogues of forms (1.1.1) - (1.4) by type (2.1*).

For example, $\begin{array}{ccccc} g_1\{R\} & g_2\{R\} & & g_n\{R\} \\ v_{11} & v_{21} & \ldots & v_{n1} \\ \mu_{11} & \mu_{21} & & \mu_{n1} \\ \{R\}_1 & \{R\}_2 & \ldots & \{R\}_n \end{array}$ PrSprt $\begin{array}{ccccc} \{R\}_1 & \{R\}_2 & & \{R\}_n \\ v_{11} & v_{21} & \ldots & v_{n1} \\ \mu_{11} & \mu_{21} & & \mu_{n1} \\ g_1\{R\} & g_2\{R\} & \ldots & g_n\{R\} \end{array}$ is the capacity in itself of the second type if $g_j\{R\}$ is a program capable of generating $\{R\}_j$.

The example of self-program of the first type is



PrffSCprt-coding: 1) fuzzy set A_i to fuzzy set B_i, fuzzy set D_i from fuzzy set C_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, …, n; j = 1, 2, …, m). For example, $\begin{matrix} C_1 & C_2 & \dots C_n \\ v_{11} & v_{21} & v_{n1} \\ \mu_{11} & \mu_{21} & \dots \mu_{n1} \\ D_1 & D_2 & \dots D_n \end{matrix}$

$$\text{PrffSCprt} \begin{matrix} A_1 & A_2 & \dots A_n \\ v_{11} & v_{21} & v_{n1} \\ \mu_{11} & \mu_{21} & \dots \mu_{n1} \\ B_1 & B_2 & \dots B_n \end{matrix}.$$

There are PrffSCprt -coding, PrffSCprt-translation, PrffSCprt-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 continuous etc. For example, PrffSCprt $\begin{matrix} A_1 & A_2 & \dots A_n \\ v_{11} & v_{21} & v_{n1} \\ \mu_{11} & \mu_{21} & \dots \mu_{n1} \\ A_1 & A_2 & \dots A_n \end{matrix}$ or $\begin{matrix} A_1 & A_2 & \dots A_n \\ v_{11} & v_{21} & v_{n1} \\ \mu_{11} & \mu_{21} & \dots \mu_{n1} \\ A_1 & A_2 & \dots A_n \end{matrix}$ PrffSCprt

$$\begin{matrix} A_1 & A_2 & \dots A_n \\ v_{11} & v_{21} & v_{n1} \\ \mu_{11} & \mu_{21} & \dots \mu_{n1} \\ B_1 & B_2 & \dots B_n \end{matrix} \text{ etc.}$$

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 PrffSCprt-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 PrffSCprt -Networks and series-parallel. Codes from a conventional type computer system will be used via PrffSCprt -connectors in PrffSCprt - coding, for example:

$$\begin{array}{cccc} activation & activation & \cdots & activation \\ v_{11} & v_{21} & \cdots & v_{n1} \\ \mu_{11} & \mu_{21} & \cdots & \mu_{n1} \\ w_1 & w_2 & \cdots & w_n \end{array} \text{ PrffSCprt(t) } \begin{array}{ccccc} w_1 & w_2 & & w_n \\ v_{11} & v_{21} & \cdots & v_{n1} \\ \mu_{11} & \mu_{21} & \cdots & \mu_{n1} \\ activation & activation & & ...activation \end{array}, \text{ } w_j =$$

$(\text{UHF AC})_j(t) := Q_j(t)$, , j = 1, 2, …, n, UHF AC field activation is used.

Consider the fuzzy dynamic PrffSCprt and $\text{PrffS}_3\text{Cf(t)}$ programming:

1. The process of simultaneous fuzzy p-assignment of the fuzzy expressions $\widetilde{p(t)}$ $=(p_1(t)|\mu_{\widetilde{p(t)}}(p_1(t)), p_2(t)|\mu_{\widetilde{p(t)}}(p_2(t)), \ldots, p_n(t)|\mu_{\widetilde{p(t)}}(p_n(t)))$ to the variables $\widetilde{x(t)}=(x_1(t)|$ $\mu_{\widetilde{x(t)}}(x_1(t)), x_2(t)|\mu_{\tilde{x}(t)}(x_2(t)), \ldots, x_n(t)|\mu_{\widetilde{x(t)}}(x_n(t)))$ is implemented through

$$\begin{array}{cccc} x_1(t) & x_2(t) & \cdots & x_n(t) \\ v_{11} & v_{21} & \cdots & v_{n1} \\ \mu_{11} & \mu_{21} & \cdots & \mu_{n1} \\ := p_1(t) & := p_2(t) & \ldots & :=p_n(t) \end{array} \text{ PrffSCprt } \begin{array}{ccccc} x_1(t) := & x_2(t) := & & x_n(t) := \\ v_{11} & v_{21} & \cdots & v_{n1} \\ \mu_{11} & \mu_{21} & \cdots & \mu_{n1} \\ p_1(t) & p_2(t) & \ldots & p_n(t) \end{array} .$$

2. The process of simultaneous check the fuzzy set of conditions $\widetilde{g(t)}=(g_1(t)|\mu_{\widetilde{g(t)}}$ $(g_1(t)), g_2(t)|\mu_{\tilde{g}(t)}(g_2(t)), \ldots, g_n(t)|\mu_{\widetilde{g(t)}}(g_n(t)))$ for the fuzzy set of expressions $\widetilde{B(t)}$ $=(B_1(t)|\mu_{\widetilde{B(t)}}(B_1(t)), B_2(t)|\mu_{\tilde{B}(t)}(B_2(t)), \ldots, B_n(t)|\mu_{\widetilde{B(t)}}(B_n(t)))$ and

solution of task Q_j by D_j simultaneously is implemented through

$$\begin{array}{cccc} task\ Q_1 & task\ Q_2 & & task\ Q_n \\ v_{21} & v_{22} & \cdots & v_{2n} \\ \mu_{21} & \mu_{22} & & \mu_{2n} \\ h_1 & h_2 & \ldots & h_n \end{array} \text{ PrffSCprt } \begin{array}{ccccc} IF\{B_1(t)g_1(t)\}\ then & IF\{B_2(t)g_2(t)\}\ then & & IF\{B_n(t)g_n(t)\}\ then \\ v_{11} & v_{21} & \cdots & v_{n1} \\ \mu_{11} & \mu_{21} & \cdots & \mu_{n1} \\ w_1(t) & w_2(t) & \ldots & w_n(t) \end{array},$$

where $w(t) = (w_1(t), w_2(t), \ldots, w_n(t))$. can be any *solution of task Q_j by D_j* with measure of fuzziness μ_{2j}, j = 1, 2, …, n.

3.Similarly for loop operators and others.

$PrffS_3Cf(t)$– 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 PrffSCprt(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 analogues of forms (1.1.1) - (1.4) by type (2.1*).

Consider PrfftSCpr-program operators. The ideology of PrftSCpr and $Prft_{S_4Cf}$ is parallel analogue of ft_{S_4f} [14] can be used for programming. Here are some of the PrftSCpr -program operators.

1. Simultaneous fuzzy expelling assignment of the fuzzy expressions $\tilde{p}=(p_1|\mu_{\tilde{p}}(p_1), p_2|\mu_{\tilde{p}}(p_2), \ldots, p_n|\mu_{\tilde{p}}(p_n))$ from the variables $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \ldots, x_n|\mu_{\tilde{x}}(x_n))$. It's implemented through

$$\begin{matrix} x_1 := & x_2 := & & x_n := \\ v_{11} & v_{21} & \ldots & v_{n1} \\ \mu_{11} & \mu_{21} & \ldots & \mu_{n1} \\ p_1 & p_2 & \ldots & p_n \end{matrix} \text{ PrffSCprt.}$$

2. Similarly for loop operators and others.

$Prfft_{S_4Cf}$ – software operators will differ only just because aggregates $\{\tilde{g}\},\{\tilde{p}\},\{\tilde{B}\},\{\tilde{x}\}$ will be formed from corresponding PrfftSCpr program operators in form (1.1) for more complex operators in forms (1.1.1) – (1.4) , (2.1*) and analogues of forms (1.1.1) - (1.4) by type (2.1*).

Consider hierarchical PrfftSCpr-program operator

$$\begin{matrix} C_1(t) & C_2(t) & \ldots & C_m(t) \\ v_{12} & v_{22} & \ldots & v_{m2} \\ \mu_{12} & \mu_{22} & \ldots & \mu_{m2} \\ D_1(t) & D_2(t) & \ldots & D_m(t) \end{matrix} \text{ PrffSCrt}_2\text{(t)} =$$

$$\left\{ \begin{matrix} \sum_{i=1}^{m} Q_i(t) + & \begin{matrix} \{\} & \{\} & \ldots & \{\} \\ v_{12} & v_{22} & & v_{m2} \\ \mu_{12} & \mu_{22} & \ldots & \mu_{m2} \end{matrix} & \text{PrffSCrt(t)} \\ D_1(t) - D_1(t) \cap C_1(t) \quad D_2(t) - D_2(t) \cap C_2(t) \quad \ldots D_m(t) - D_2(t) \cap C_m(t) & & \\ \sum_{i=1}^{m} (C_i(t) - D_i(t) \cap C_i(t)) - (D_i(t) - D_i(t) \cap C_i(t)) & & \end{matrix} \right\}$$

, where $Q_i(t)$ is oself-(fuzzy set) for $(D_i(t) \cap C_i(t))$ [14].

Consider the PrtffSCpr(t) and $Prfft(t)_{S_4f}$ programming at time t.

1. The process of simultaneous assignment of the expressions $\widetilde{p(t)}=(p_1(t)|\mu_{\widetilde{p(t)}}(p_1(t)), p_2(t)|\mu_{\widetilde{p(t)}}(p_2(t)), \ldots, p_n(t)|\mu_{\widetilde{p(t)}}(p_n(t)))$ to the variables $\widetilde{x(t)}=(x_1(t)|\mu_{\widetilde{x(t)}}(x_1(t)),$

$x_2(t)|\mu_{\tilde{x}(t)}(x_2(t)), \ldots, x_n(t)|\mu_{\widetilde{x(t)}}(x_n(t)))$ is implemented through

$$\begin{matrix} x_1(t) := & x_2(t) := & & x_n(t) := \\ v_{11} & v_{21} & \cdots & v_{n1} \\ \mu_{11} & \mu_{21} & \ldots & \mu_{n1} \\ p_1(t) & p_2(t) & \ldots & p_n(t) \end{matrix} \text{ PrffSCprt(t).}$$

2. Similarly for loop operators and others.

$Prfft(t)_{S_4Cf}$ – software operators will differ only in that the aggregates $\widetilde{x(t)}$,$\widetilde{p(t)}$, $\widetilde{B(t)}$,$\widetilde{g(t)}$ will be formed from corresponding processes PrffSCprt(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 analogues of forms (1.1.1) - (1.4) by type (2.1*).

Consider PrffSCrt(t)- program operators

$$\{ {}^{q}\begin{pmatrix} u & u & u & u \\ v_1 & v_2 & v_1 & v_2 \\ \mu_1 & \mu_2 & \mu_1 & \mu_2 \\ u & u & u & u \end{pmatrix}_{PrffSCpr} W_q fSprt_q^{E_q}\begin{pmatrix} u & u & u & u \\ v_1 & v_2 & v_1 & v_2 \\ \mu_1 & \mu_2 & \mu_1 & \mu_2 \\ u & u & u & u \end{pmatrix}_{PrffSCpr}, \; fSprt_{d_r\left(E_{in}{}^{l}d_r\right)}^{\{E^{ex}{}_{l}{}^{d}{}_{r}\}} \}$$

$fSprt_{t_0}$. —program

structure example, where the assemblage point d_r is the cursor, it is quite complex ffself—program.

Remark. Energy with measure of fuzziness μ_1, μ_2 of a living organism other than humans:

fPrff(r, u(E_q), μ_1, μ_2) =

$$\{ {}^{q}\begin{pmatrix} u & u & u & u \\ v_1 & v_2 & v_1 & v_2 \\ \mu_1 & \mu_2 & \mu_1 & \mu_2 \\ u & u & u & u \end{pmatrix}_{PrffSCpr} W_q Sprt_q^{E_q}\begin{pmatrix} u & u & u & u \\ v_1 & v_2 & v_1 & v_2 \\ \mu_1 & \mu_2 & \mu_1 & \mu_2 \\ u & u & u & u \end{pmatrix}_{PrffSCpr}, \; fSprt_{d_r\left(E_{in}{}^{l}d_r\right)}^{\{E^{ex}{}_{l}{}^{d}{}_{r}\}} \}$$

$fSprt_{t_0}$ $(**_{F.3})$.

Energy with measure of fuzziness μ_1, μ_2 of a living organism of a person:

fPrhff(r, u(E_q), μ_1, μ_2) =

$$\{ {}^{q}\begin{pmatrix} u & u & u & u \\ v_1 & v_2 & v_1 & v_2 \\ \mu_1 & \mu_2 & \mu_1 & \mu_2 \\ u & u & u & u \end{pmatrix}_{PrffSCpr} W_q Sprt_q^{E_q} \begin{pmatrix} u & u & u & u \\ v_1 & v_2 & v_1 & v_2 \\ \mu_1 & \mu_2 & \mu_1 & \mu_2 \\ u & u & u & u \end{pmatrix}, fSprt_{d_r(self(E_{in}l^{d_r}))}^{\{E^{ex}l^{d_r}\}} \}$$

$fSprt_{t_0}$ (***$_{F.3}$).

$\begin{matrix} u & u & u & u \\ v_1 & v_2 & v_1 & v_2 \\ \mu_1 & \mu_2 & \mu_1 & \mu_2 \\ u & u & u & u \end{matrix}_{PrffSCpr}$ -internal energy with measure of fuzziness μ_1, μ_2 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.

(**$_{F.3}$), (***$_{F.3}$) can be interpreted as PrffSCrt- program operators.

3.14 FSUprt and FS$_3$Uf programming and their pself-type

The ideology of FSUprt and FS$_3$Uf can be used for programming. Here are some of the FSUprt programming operators:

1. Simultaneous random p-assignment of the expressions $\tilde{r}$=($r_1|\mu_{\tilde{r}}(r_1)$, $r_2|\mu_{\tilde{r}}(r_2)|_{,\dots,}r_n|\mu_{\tilde{r}}(r_n)|$) 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 ${}^{q}_{p}\overleftarrow{FSUprt}^{\{\{\tilde{x}\}:=\tilde{r}\}}_{\{\{\tilde{x}\}:=\tilde{r}\}}{}_{q}$.

2. Simultaneous random p-checking the fuzzy set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2)|,\ldots,g_n|\mu_{\tilde{g}}(g_n)|)$ for the fuzzy set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2)|,\ldots,x_n|\mu_{\tilde{B}}(B_n)|)$ Implemented via $IF\left\{\left\{\tilde{B}\right\}\{\tilde{g}\}\right\}_{p}^{q}$ *then* $\tilde{Q}$ FSUprt

$IF\left\{\left\{\tilde{B}\right\}\{\tilde{g}\}\right\}_{q}^{p}$ *then* $\tilde{Q}$. where $\tilde{Q}$ can be anything.

3. Similarly for loop operators and others.

FS_3Ufp – fuzzy software operators will differ only in that the aggregates $\{\tilde{x}\},\{\tilde{r}\},\left\{\tilde{B}\right\},\{\tilde{g}\}$ will be formed from the corresponding FSUprt program operators in p-analogue of form (1.1) or p-analogue of forms (1.1.1) - (1.4), (2.1*) and analogues 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.

3.15 The usage of FSUprt-elements for networks

A. Galushkin's comprehensive monograph [20] 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 random containment operators, which, although they can be interpreted as the result of some correspondence operators, are not themselves correspondence operators [1]. The random 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 FSUprt - networks. FSUprt-networks(FSUmnsprt) are a FSUprt -structure that can be built for the required weights. FSUprt -OS (FSUprtoperating system) uses FSUprt - coding and FSUprt -translation. In the first one, coding is carried out through a 2-dimensional matrix-row (a, b), where the random number b is the code of the action, and the random number a is the code of the object of this action. FSUprt - coding (or rself- (random coding)) is implemented through a random matrix

consisting of 2 columns (in the continuous case, two intervals of random numbers). Here, the source encoding is used for all matrix rows simultaneously. FSUprt - translation is carried out by inversion. In this case, rself- (fuzzy coding) and rself- (fuzzy translation) will be more stable. The target weights $q_i(t)$ in $\widetilde{w}(t)$ $p(t)$ FSUprt(t) $\widetilde{x}(t)\widetilde{q}(t)$ $p(t)$ are chosen for necessary tasks. We will not touch on $\widetilde{x}(t)\widetilde{q}(t)$ $\widetilde{w}(t)$ the issues of applications, or network optimization. They are described in detail by Galushkin [20]. 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 FSUprt(designation - mnFSUprt) for simple information processing. More complex executing programs are used for mnFSUprt nodes.

Threshold element FSUprt - b p_2 FSUprt(t) $\{a\tilde{x}\}$ p_1 $\{qy\}$ b , b- artificial neurons of type FSUprt (designation - mnFSUprt) , $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2)|,\ldots,x_n|\mu_{\tilde{x}}(x_n)|)$ is the fuzzy set of values of the initial signals, $a=(a_1,a_2,\ldots,a_n)$ are the weights of Sit-synapses and $\tilde{y}$ $=(y_1|\mu_{\tilde{y}}(y_1), y_2|\mu_{\tilde{y}}(y_2)|,\ldots,y_n|\mu_{\tilde{y}}(y_n)|)$ is the fuzzy set of values of the output signals with weights $q=(q_1,q_2,\ldots,q_n)$.. The first level of mnFSUprt consists of simple mnFSUprt. The second level of mnFSUprt consists of D p FSUprt mnFSUprt p D – FSUprt-node of mnFSUprt in range D, D- fuzzy mnFSUprt capacity for mnFSUprt node. The third level of mnFSUprt consists of D p D p FSUprt mnFSUprt p D mnFSUprt FSUprt D p D p FSUprt mnFSUprt p D - FSUprt ^{2}- mnFSUprt node of mnFSUprt in range D, thus D becomes fuzzy capacity of itself in itself as an element for mnFSUprt. For our networks, it is sufficient to use FSUprt ^{2}- nodes of mnFSUprt, but rself-level is higher in living organisms, particularly FSUprt ^{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 3.15. 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. mnFSUprt contains $\overset{\text{mnFSUprt}}{p}_{\text{FSUprt}}\overset{\{\text{freprograms}\}}{p}$, freprogram –executing program in $\overset{\{\text{freprograms}\}}{\text{mnFSUprt}}$ FSUprt - OS. FSUprt -OS (or Rself-OS) is based on FSUprt -assembly language (or FSUelf-assembly language), which is based on assembly language through FSUprt -approach in turn, if the base of elements of FSUprt -networks is sufficient. The freprograms are in FSUprt -programming environments (or rself- fuzzy programming environments), but this question and FSUprt -networks base will be considered in the following monographs. In particular, freprograms may contain FSUprt - programming operators. In mnFSUprt cores, the constant memory FSUprt with correspondent freprograms depending on mnFSUprt.

The OS (operating system) and the principles and modes of operation of the FSUprt -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 FSUprt– physics and special neural networks with artificial neurons operating in normal and FSUprt -modes. Let's denote this model through FSUmnsprt. To do this, it’s proposed to use mnFSUprt of different levels; for example, for the usual mode, mnFSUprt 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 FSUprt –mode with the desired "target weight" is realized in this section, etc., to the center. In the case of a monster during the test, FSUmnsprt is activated with the desired "target weight." Here are realized other tasks also. To reach the rself-energy level, the mode $\overset{\text{SUmnsprt}}{p}_{\text{FSUprt}}\overset{\text{SUmnsprt}}{p}$ is used. In

normal mode, it’s planned to carry out the movement of FSUmnsprt 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 FSUmnsprt 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 FSUmnsprt. FSUmnsprt is represented by a neural network that extends from the center of one of the main clusters of FSUprt-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 FSUmnsprt 's actions is below the operator's cab. In FSUprt– mode, the entire network or its sections are FSUprt – activated to perform specific tasks, in particular, with "target weights." In the target, block used FSUprt -coding, FSUprt -translation for activation of all networks to "target weights" simultaneously, then –the reset of this FSUprt -coding after activation. Unfortunately, triodes are not suitable for FSUprt -neural networks. In the most primitive case, usual separators with corresponding resistances and cores for freprograms may be used instead triodes since there is no necessity to unbend the alternating current to direct. The FSUprt-operative memory belt is disposed around a central core of FSUmnsprt. There are FSUprt -coding, FSUprt -translation, and FSUprt-realize of feprograms and the programs from the archives without extraction, FSUprt-coding and FSUprt-translation may be used in high-intensity, ultra-short optical pulses laser of Nobel laureates 2018-year Gerard Mourou, Donna, Strickland. FSUprt– structure or an freprogram if one is present of needed «target weight» are taken in target block at FSUprt– activation of the networks. $\mu_{activation}^{FSUprt}(SUmnsprt,g)$ derives SUmnsprt to the ruself-level boundary $\mu_{activation}(SUmnsprt,g)$ with target weight g.

It's used an alternating current of above high frequency and ultra-violet light, which can work with FSUprt– structures in FSUprt –modes by its nature to activate the networks or some of its parts in FSUprt –modes and locally using FSUprt –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."

3.16 RANDOM PROGRAM OPERATORS FSUprt, fftprS, FSU^{1}epr, FSUeprt$_{1}$ and their pself-type

Here it is supposed to use a symbiosis of parallel actions and conventional calculations through sequential actions [1]. This must be done through FSUprt-Networks - random analogue of Sit-Networks [15] in one of the central departments of which a conventional computer system is located. The parallel processor is itself rfeprogram - random analogue of eprogram [15] with direct parallel computing not through serial computing.

Using conventional coding by a computer system, through a Target-block with a FSUprt -program operator - $\begin{matrix} activation \\ p \\ Ag \end{matrix}$ FSUprt $\begin{matrix} Ag \\ p \\ activation \end{matrix}$ with probabilities p, it will be possible to obtain the random execution with probabilities p of a parallel random action A with the desired target weight g or the execution of a parallel action A with the desired random 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 FSUprt-coding and FSUprt-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 rfprogram of operator $\begin{matrix} activation \\ p \\ \{\text{UHF AC } := Q\} \end{matrix}$ FSUprt $\begin{matrix} \{\text{UHF AC } := Q\} \\ p \\ activation \end{matrix}$ with probabilities p, 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 random fprogram operators:

7) random fSprt-program operators (designation FSUprt-program operators)
8) random ftprS-program operators (designation FtprS-program operators)
9) random fS^{1}epr - program operators (designation FSU^{1}epr -program operators)
10) random Seprt$_1$- program operators (designation FSUeprt$_1$-program operators)

One example is pattern recognition: $\underset{p}{\text{FSUprt}}$

if $\underset{B}{\text{FSUprt}}$ *image archive* $\ni \underset{B}{\text{FSUprt}}\overset{q}{p}$ *then Name of q* .

The example of FSUprt-program is $\underset{p}{\text{FSUprt}}$

$$\{\underset{p}{\text{FSUprt}}\{\{r\} := \{\mathsf{a}(x)\}\}, \underset{p}{\text{FSUprt}}\{IF\{\{B\}\{f\}\} \text{ then } Q\}, \underset{Q}{\text{FSUprt}}p\}.$$

Consider a third type of random capacity in itself - random analogue of capacity in itself. For example, based on $\underset{\tilde{x}}{\overset{q}{p}}\text{FSUprt}\underset{q}{\overset{\tilde{x}}{p}}$, where $\tilde{x} = (x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \ldots, x_n|\mu_{\tilde{x}}(x_n))$,

i.e. n - elements at one point, we can consider the capacity $FS_3Uf\tilde{x}p$ (in itself with m elements from $\tilde{x}$, m<n, which is formed according to the p-analogue of form (1.1),

that is, the structure $\underset{q}{\text{FSUprt}}\overset{\tilde{x}}{p}$ contains only m elements or for more complex

operators in p-analogue of forms (1.1.1) - (1.4), (2.1*) and analogues of forms (1.1.1) - (1.4) by type (2.1*)..

Here are some of the random fSUprt-program operators.

1. Simultaneous random p-assignment with fuzziness m of the expression $s\ \tilde{r}=(r_1|\mu_{\tilde{r}}(r_1), r_2|\mu_{\tilde{r}}(r_2), \ldots, r_n|\mu_{\tilde{r}}(r_n))$ to the variables $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \ldots, x_n|\mu_{\tilde{x}}(x_n))$. This is implemented via $\underset{\{\{\tilde{x}\}:=\{r\}\}}{\overset{w\ p}{\text{FSUprt}}}$ $\underset{p\ w}{\overset{\{\{\tilde{x}\}:=\{r\}\}}{\text{FSUprt}}}$.
2. Simultaneous random p-checking with probabilities p by the fuzzy set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2), \ldots, g_n|\mu_{\tilde{g}}(g_n))$ for the fuzzy set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2), \ldots, B_n|\mu_{\tilde{B}}(B_n))$ Implemented via $IF\left\{\left\{\tilde{B}\right\}\{\tilde{g}\}\right\}\ then\ \tilde{Q}$ $\underset{p\ w}{\overset{w\ p}{\text{FSUprt}}}$ $IF\left\{\left\{\tilde{B}\right\}\{\tilde{g}\}\right\}\ then\ \tilde{Q}$ where $\tilde{Q}$ can be anything.
3. Similarly for random loop operators and others.

FS_3Uf- random software operators will differ only just because aggregates $\tilde{x},\tilde{r},\ \tilde{B},\tilde{g}$ will be formed from corresponding FSUprt-program operators in p-analogue of form (1.1), for more complex operators in p-analogue of forms (1.1.1) - (1.4), (2.1*) and analogues of forms (1.1.1) - (1.4) by type (2.1*).

For example, $\underset{\{R\}}{\overset{g\{R\}}{p}}$ FSUprt $\underset{g\{R\}}{\overset{\{R\}}{p}}$ is the random capacity with probabilities p in itself of the second type if $g\{R\}$ is a rfprogram capable of random generating $\{R\}$ with probabilities p.

The example of self-rfprogram of the first type is

$$\underset{p\ w}{\overset{p\ w}{\text{FSUprt}}}\{\ \underset{p\ w}{\overset{\{\{r\}:=\{\mathsf{a}(x)\}\}}{\text{FSUprt}}}\ ,\ \underset{p\ w}{\overset{IF\{\{B\}\{f\}\}\ then\ Q}{\text{FSUprt}}}\ ,\ \underset{Q}{\overset{Q}{\text{FSUprt}}}\ p\ \}.$$

The example of FSUprt-program for FSUmnsprt:

$\underset{B}{\overset{q:=}{\text{FSUprt}}}$ - random assigning random q to fuzzy B with probabilities p.

$\underset{g}{\overset{tw:=}{\text{FSUprt}}}$ - random assigning target weight tw to fuzzy g with probabilities p.

$f\,\mathrm{fSmnsprt}\ activation \underset{p}{\mathrm{FSUprt}} \{q\}w$ $\{q\}w \underset{p}{\mathrm{FSUprt}} f\,\mathrm{fSmnsprt}\ activation$ - FSUmnsprt random *activation* for fuzzy $\{q\}w$ with probabilities p.

FSUprt-coding.

FSUprt-coding with probabilities p: 1) fuzzy set A to fuzzy set B, fuzzy set D from fuzzy set C, 2) fuzzy set A to a point q, where the elements of the fuzzy sets A, B can be continuous. For example, $\underset{D}{p}\,\mathrm{FSUprt}\,\underset{B}{p}$.

There are FSUprt-coding, FSUprt-translation, FSUprt-realize of rfeprograms and of the rfprograms from the archives without extraction theirs

FSUelf-coding.

FSUelf-coding with probabilities p: 1) fuzzy set A to set fuzzy A, i.e. fuzzy A on itself 2) fuzzy set A to a point q in p-analogue of form (1.1), where the elements of the fuzzy sets A, B can be continuous. For example, $\underset{A}{p}\,\mathrm{FSUprt}\,\underset{A}{p}$ or $\underset{B}{p}\,\mathrm{FSUprt}\,\underset{A}{p}$ etc.

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 FSUprt-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 FSUprt-Networks and series-parallel. Codes from a conventional type computer system will be used via FSUprt -connectors in FSUprt - coding, for example:

$activation \underset{p}{\mathrm{FSUprt}} \{\mathrm{UHF\ AC}\ := Q\}$ $\{\mathrm{UHF\ AC}\ := Q\} \underset{p}{\mathrm{FSUprt}} activation$. UHF AC field activation is used.

Dynamic FSUprt and $FS_3Uf(t)$ programming [1].

The ideology of dynamic FSUprt and $FS_3Uf(t)$ can be used for programming:

1. The process of simultaneous random p-assignment of the fuzzy expressions $r(\widetilde{t})=(r_1(t)|\mu_{r(\widetilde{t})}(r_1(t)), r_2(t)|\mu_{r(\widetilde{t})}(r_2(t))|,\dots,r_n(t)|\mu_{r(\widetilde{t})}(r_n(t)))$ to the fuzzy variables $x(\widetilde{t})=(x_1(t)|\mu_{x(\widetilde{t})}(x_1(t)), x_2(t)|\mu_{\tilde{x}(t)}(x_2(t))|,\dots,x_n(t)|\mu_{x(\widetilde{t})}(x_n(t)))$. This is implemented via

$$\begin{matrix} w(\widetilde{t}) \\ p(t) \\ \{\{x(\widetilde{t})\} \widetilde{:=} \{r(\widetilde{t})\}\} \end{matrix} \text{SUprt}(t) \begin{matrix} \{\{x(\widetilde{t})\} \widetilde{:=} \{r(\widetilde{t})\}\} \\ p(t) \\ w(\widetilde{t}) \end{matrix}.$$

2. The process of simultaneous random p-checking the fuzzy set of conditions $g(\widetilde{t})=(g_1(t)|\mu_{g(\widetilde{t})}(g_1(t)), g_2(t)|\mu_{g(\widetilde{t})}(g_2(t))|,\dots,g_n(t)|\mu_{g(\widetilde{t})}(g_n(t))|)$ for the fuzzy set of expressions $B(\widetilde{t})=(B_1(t)|\mu_{B(\widetilde{t})}(B_1(t)), B_2(t)|\mu_{\tilde{B}(t)}(B_2(t))|,\dots,B_n(t)|\mu_{B(\widetilde{t})}(B_n(t))|)$. Implemented via

$$\begin{matrix} w(\widetilde{t}) \\ p(t) \\ IF\{\{B(\widetilde{t})\}\{\tilde{g}(\widetilde{t})\}\} \text{ then } Q(\widetilde{t}) \end{matrix} \text{FUprt}(t) \begin{matrix} IF\{\{B(\widetilde{t})\}\{\tilde{g}(\widetilde{t})\}\} \text{ then } Q(\widetilde{t}) \\ p(t) \\ w(\widetilde{t}) \end{matrix} \text{ where}$$

$Q(\widetilde{t})$ can be anything.

3. Similarly for fuzzy loop operators and others.

$FS_3Uf(t)_{p(t)}$– fuzzy rsoftware operators will differ only just because aggregates $x(\widetilde{t}), r(\widetilde{t}), B(\widetilde{t}), g(\widetilde{t})$ will be formed from corresponding processes by FSUprt-program operators in p-analogue of form (1.1) for more complex operators in p-analogue of forms (1.1.1) - (1.4), (2.1*) and analogues of forms (1.1.1) - (1.4) by type (2.1*).

FtprS-program operators.

The ideology of FtprSU and Ft_{S_4Uf} - random analogues of tS and t_{S_4f} from [10] can be used for programming. Here are some of the FtprSU -program operators.

1. Simultaneous expelling assignment of the expressions $\tilde{r}=(r_1|\mu_{\tilde{r}}(r_1), r_2|\mu_{\tilde{r}}(r_2), \dots, r_n|\mu_{\tilde{r}}(r_n))$ 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 via $\begin{matrix} \tilde{x} \\ p \\ =:\tilde{r} \end{matrix}$ FSUprt.

2. Simultaneous expelling check the fuzzy set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2), \ldots, g_n|\mu_{\tilde{g}}(g_n))$ for the fuzzy set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2), \ldots, B_n|\mu_{\tilde{B}}(B_n))$. It's implemented through $IF\left\{\left\{\tilde{B}\right\}\overset{w}{\underset{p}{}}\left\{\tilde{g}\right\}\right\}$ *then* $\tilde{Q}^{\text{FSUprt}}$ where Q can be any.
3. Similarly for loop operators and others.

Ft_{S_4Uf} – random software operators will differ only just because aggregates $\tilde{x},\tilde{r}, \tilde{B},\tilde{g}$ will be formed from corresponding FtprSU program operators in p-analogue of form (1.1) for more complex operators in p-analogue of forms (1.1.1) - (1.4), (2.1*) and analogs of forms (1.1.1) - (1.4) by type (2.1*). Consider hierarchical FtprSU -program operator

$$\overset{B}{\underset{A}{p}}\text{FSUprt} = \left\{\begin{matrix} D + \overset{\{\}}{\underset{A - A \cap B}{p}}\text{FSUprt} \\ (B - A \cap B) \end{matrix}\right\}, \text{ where D is oself-(fuzzy set) for fuzzy } (A \cap B).$$

Dynamic FtprSU and Ft_{S_4Uf} programming at time q.

The ideology of FtprSU and Ft_{S_4Uf} can be used for dynamic programming. Here are some of the FtprSU(t)- dynamic programming operators.

1. The process of simultaneous random expelling assignment of the expressions $\widetilde{r(t)}=(r_1(t)|\mu_{\widetilde{r(t)}}(r_1(t)), r_2(t)|\mu_{\widetilde{r(t)}}(r_2(t)), \ldots, r_n(t)|\mu_{\widetilde{r(t)}}(r_n(t)))$ from the variables $\widetilde{x(t)}$ $=(x_1(t)|\mu_{\widetilde{x(t)}}(x_1(t)), x_2(t)|\mu_{\widetilde{x(t)}}(x_2(t)), \ldots, x_n(t)|\mu_{\widetilde{x(t)}}(x_n(t)))$.is implemented through $\overset{\tilde{x}(t)}{\underset{p(t)}{}}\text{FSUprt(t)}.$ $= :\tilde{r}(t)$

2. The process of simultaneous random expelling check the fuzzy set of conditions $\widetilde{g(t)}=(g_1(t)|\mu_{\widetilde{g(t)}}(g_1(t)), g_2(t)|\mu_{\widetilde{g(t)}}(g_2(t)), \ldots, g_n(t)|\mu_{\widetilde{g(t)}}(g_n(t)))$ for the fuzzy set of expressions $\widetilde{B(t)}=(B_1(t)|\mu_{\widetilde{B(t)}}(B_1(t)), B_2(t)|\mu_{\widetilde{B(t)}}(B_2(t)), \ldots, B_n(t)|\mu_{\widetilde{B(t)}}(B_n(t)))$ is

implemented through $IF\left\{\left\{\tilde{B}(t)\right\}\left\{\tilde{g}(t)\right\}\right\}\ then\ \tilde{Q}(t)$ $\overset{w(t)}{\underset{p(t)}{\text{FSUprt(t)}}}$, where Q(t) can be any.

3.Similarly for random loop operators and others.

Dynamic Ft_{S_4Uf} – random software operators will differ only just because aggregates $\widetilde{x(t)}$,$\widetilde{r(t)}$, $\widetilde{B(t)}$,$\widetilde{g(t)}$ will be formed from corresponding processes fftprS(t) for above mentioned programming operators through p-analogue of form (1.1) for more complex operators in p-analogue of forms (1.1.1) - (1.4), (2.1*) and analogues of forms (1.1.1) - (1.4) by type (2.1*). Consider hierarchical dynamic FtprSU(t)-program operator:

$$\overset{B(q)}{\underset{A(q)}{p(q)\,\text{FSUprt(q)}}} = \left\{ Ft(q)_{S_1Uf(A(q)\cap B(q))} + \overset{\{\}}{\underset{A(q)-A(q)\cap B(q)}{p(q)}}\ \text{FSUprt(q)} \left(B(q)-A(q)\cap B(q)\right) \right\},$$

FSU[1]epr -program operators (form $\underset{D}{\overset{B}{p_2}}F\text{SU}^1\text{prt}\underset{B}{\overset{A}{p_1}}$ - random analogue of ${}^{B}_{D}S^1t^{A}_{B}$ [6])**)**

For example, $\underset{\{=:\{p(t)\}\}}{\overset{\{\text{a}(t)\}}{p_2(t)}}\ \text{SU}^1\text{prt}\ \underset{\{\text{a}(t)\}}{\overset{IF\{\{B(t)\}\{f(t)\}\}\ then\ Q(t)}{p_1(t)}}$.

Consider hierarchical dynamic FSU[1]epr-program operator: (form

$$\underset{A}{\overset{B}{p_2}}\text{FS}^1\text{U}\text{prt}\underset{B}{\overset{A}{p_1}}*$$

$$\underset{B}{\overset{A}{p_2}}\ \text{FS}^1\text{U}\text{prt}\underset{B}{\overset{A}{p_1}}\ ).$$

$$\underset{D-A}{\overset{B}{p_2}}\ \text{FS}^1\text{U}\text{prt}\underset{B}{\overset{A}{p_1}}$$

FSUeprt$_1$- program operators (form $\underset{D}{\overset{C}{p_2}}F\text{SU}_1\text{prt}\underset{B}{\overset{A}{p_1}}$ - random analogue of ${}^{C}_{D}S_1t^{A}_{B}$ [11])**).**

$\mathrm{FSUprt}\left\{ \begin{matrix} \begin{pmatrix} a & a \\ p_1\mathrm{SUrt}p_1 & \\ a & a \end{pmatrix} W_q f Sprt^{E_q}_{q\begin{pmatrix} a & a \\ p_1\mathrm{SUrt}p_1 & \\ a & a \end{pmatrix}}, & f Sprt^{\{E^{ex}l^{d_r}\}}_{d_r\left(E_{in}l^{d_r}\right)} \end{matrix} \right\}_{\substack{p \\ t_0}}$ —rfprogram structure example,

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

$\mathrm{FSUprt}\left\{ \begin{matrix} q\begin{pmatrix} a & a \\ p_1\mathrm{SUrt}p_1 & \\ a & a \end{pmatrix} W_q f Sprt^{E_q}_{q\begin{pmatrix} a & a \\ p_1\mathrm{SUrt}p_1 & \\ a & a \end{pmatrix}}, & f Sprt^{\{E^{ex}l^{d_r}\}}_{d_r\left(E_{in}l^{d_r}\right)} \end{matrix} \right\}_{\substack{p \\ t_0}}$ can be interpreted as a rfprogram operator.

$\begin{matrix} a & a \\ p_1\mathrm{SU}_1\mathrm{rt}p_1 & \\ a & a \end{matrix}$ can be interpreted as a $\begin{pmatrix} s_1elf \\ os_1elf \end{pmatrix}$-rprogram operator. $\begin{matrix} a & a \\ p_1\mathrm{SU}_1\mathrm{rt}p_1 & \\ a & a \end{matrix}$ -- sample $\begin{pmatrix} s_1elf \\ os_1elf \end{pmatrix}$-rprogram structure example.

Appendix

Remark. Energy of a living organism:

$$\mathrm{rfg(r,\ a}(E_q)) = \mathrm{FSUprt}\left\{ \begin{matrix} q\begin{pmatrix} a & a \\ p_1\mathrm{SUrt}p_1 & \\ a & a \end{pmatrix} W_q f Sprt^{E_q}_{q\begin{pmatrix} a & a \\ p_1\mathrm{SUrt}p_1 & \\ a & a \end{pmatrix}}, & f Sprt^{\{E^{ex}l^{d_r}\}}_{d_r\left(E_{in}l^{d_r}\right)} \end{matrix} \right\}_{\substack{p \\ t_0}} (**_{3.7}).$$

$\begin{matrix} a & a \\ p_1\mathrm{SUrt}p_1 & \\ a & a \end{matrix}$ -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, $E^{ex}l^{d_r}$ - 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.

Energy with probabilities p_1, p of a living organism of a person:

$$\text{rfpg}(r, a(E_q)) = \text{FSUprt}\left\{ {}^{q}{}_{\begin{pmatrix} a & a \\ p_1\text{SUrt} & p_1 \\ a & a \end{pmatrix}} W_q fSprt^{E_q}_{q_{\begin{pmatrix} a & a \\ p_1\text{SUrt} & p_1 \\ a & a \end{pmatrix}}}, \quad fSprt^{\{E^{ex}l^{d_r}\}}_{d_r\left(self\left(E_{in}l^{d_r}\right)\right)} \right\}_{t_0}^{p} (***_{3.7}).$$

$(**_{3.7})$, $(***_{3.7})$ can be interpreted as FSUprt - program operators.

Entire neural network as instantaneous simultaneous rFRAM in FSUprt-elements and rself- elements. $rself^{rself^{\cdots^{rself}}}$, $\text{rf1}\downarrow \text{I} \uparrow^1_{-1} rf_2{}^{\text{rf1}\downarrow \text{I}\uparrow^1_{-1} rf_2{}^{\cdots^{\text{rf1}\downarrow \text{I}\uparrow^1_{-1} rf_2}}}$,

$Fsin\infty^{Fsin\infty^{\cdots^{Fsin\infty}}}$, $Fsin\infty$ *is the* random $sin\infty$. When activated in a neural network, the entire neural network becomes a working memory. Use of rself-energy as random activation or from outside. $FQ_0 = \text{FSUprt}\ {}^{FSU_{mnSprt}}_{p}{}_{activation}\ \mu\ \text{FSUprt}\ {}^{FSU_{mnSprt}}_{p}{}_{activation} \rightarrow$ self-FrRAM,

$FQ_{00} = {}^{FSU_{mnSprt}}_{p}{}_{activation}\ \text{FSUprt}\ ffQ_0\text{prt}$, $FQ_{01} = {}^{FSU_{mnSprt}}_{p}{}_{activation}\ \text{FSUprt}\ FQ_0.FQ_0, FQ_{00}, FQ_{01}$-coding,translation,realization in rfeprograms, use FQ_0,FQ_{00},FQ_{01}-FSU_{mnSprt},FAssembler.

3.17 The usage of fSAprt-elements for networks

A. Galushkin's comprehensive monograph [20] 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 [1]. 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 fSAprt -networks. fSAprt-networks(fSAmnsprt) are a fSAprt -structure that can be built for the required weights. fSAprt -OS (fSAprt operating system) uses fSAprt -coding and fSAprt - 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. fSAprt -coding (or sel_Qf - (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. fSAprt -translation is carried out by inversion. In this case, sel_Qf - (fuzzy coding) and sel_Qf - (fuzzy translation) will be more stable. The target weights $q_i(t)$ in $\begin{matrix} w\widetilde{(t)} \\ Q^{-1}(t) \\ x\widetilde{(t)}\mathrm{q}\widetilde{(t)} \end{matrix}$ fSAprt(t) $\begin{matrix} x\widetilde{(t)}\mathrm{q}\widetilde{(t)} \\ Q(t) \\ w\widetilde{(t)} \end{matrix}$ are chosen for necessary tasks. We will not touch on the issues of applications, or network optimization. They are described in detail by Galushkin [20]. 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 fSAprt(designation - mnfSAprt) for simple information processing. More complex executing programs are used for mnfSAprt nodes. Threshold element fSAprt - $\begin{matrix} b \\ Q_2 \\ \{qy\} \end{matrix}$ fSAprt(t) $\begin{matrix} \{a\tilde{x}\} \\ Q_1 \\ b \end{matrix}$, b- artificial neurons of type fSAprt (designation - mnfSAprt) , $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2)|,\ldots,x_n|\mu_{\tilde{x}}(x_n)|)$ is the fuzzy set of values of the initial signals, $a=(a_1,a_2,\ldots,a_n)$ are the weights of fSAprt -synapses and $\tilde{y}=(y_1|\mu_{\tilde{y}}(y_1), y_2|\mu_{\tilde{y}}(y_2)|,\ldots,y_n|\mu_{\tilde{y}}(y_n)|)$ is the fuzzy set of values of the output signals with weights $q=(q_1,q_2,\ldots,q_n)$.. The first level of mnfSAprt consists of simple mnfSAprt. The

second level of mnfSAprt consists of $fSt_D^{\{\text{mnSt}\}}$ – fSAprt -node of mnfSAprt in range D, D- fuzzy capacity for mnfSAprt node. The third level of mnfSAprt consists of $\underset{\text{fSAprt}}{\overset{D}{\underset{Q^{-1}(t)}{\text{mnfSAprt}}}}$ $\underset{Q}{\overset{\text{mnfSAprt}}{\underset{D}{\text{fSAprt}}}}$ - fSAprt ^{2}- node of mnfSAprt in range D, thus D becomes fuzzy capacity of itsel$_Q$f in itsel$_Q$f as an element for mnfSAprt. For our networks, it is sufficient to use fSAprt ^{2}- nodes of mnfSAprt, but sel$_Q$f -level is higher in living organisms, particularly fSAprt ^{n}-, n≥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 3.5. 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. mnfSAprt contains $\underset{Q^{-1}(t)}{\overset{\text{mnfSAprt}}{\text{fSAprt}}}$ $\underset{\text{mnfSAprt}}{\overset{\{\text{fAeprograms}\}}{Q}}$, faeprogram –executing program in fSAprt {fAeprograms} - OS. fSAprt -OS (or sel$_Q$f -OS) is based on fSAprt -assembly language (or sel$_Q$f -assembly language), which is based on assembly language through fSAprt -approach in turn, if the base of elements of fSAprt -networks is sufficient. The faeprograms are in fSAprt -programming environments (or sel$_Q$f - fuzzy programming environments), but this question and fSAprt -networks base will be considered in the following monographs. In particular, faeprograms may contain fSAprt - programming operators. In mnfSAprt cores, the constant memory fSAprtwith correspondent faeprograms depending on mnfSAprt.

The OS (operating system) and the principles and modes of operation of the fSAprt -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 fSAprt– physics and special neural networks with artificial neurons operating in normal and fSAprt -modes. Let's denote this model through fSAmnsprt. To do this, it’s proposed to use mnfSAprt of different levels; for example, for the usual mode, mnfSAprt 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 fSAprt –mode with the desired "target weight" is realized in this section, etc., to the center. In the case of a monster during the test, fSAmnsprt is activated with the desired "target weight." Here are realized other tasks also. To reach the sel$_Q$f -energy level, the mode $Q^{-1}(t)$ fSAprt Q 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 fSAmnsprt 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 fSAmnsprt. fSAmnsprt is represented by a neural network that extends from the center of one of the main clusters of fSAprt-artificial neurons to the shell, turning into the body itsel$_Q$f. 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 fSAmnsprt 's actions is below the operator's cab. In fSAprt– mode, the entire network or its sections are fSAprt– activated to perform specific tasks, in particular, with "target weights." In the target, block used fSAprt -coding, fSAprt -translation for activation of all networks to "target weights" simultaneously, then –the reset of this fSAprt -coding after activation. Unfortunately, triodes are not suitable for fSAprt -neural networks. In the most primitive case, usual separators with corresponding resistances and cores for

faeprograms may be used instead triodes since there is no necessity to unbend the alternating current to direct. The fSAprt-operative memory belt is disposed around a central core of fSAmnsprt. There are fSAprt -coding, fSAprt -translation, and fSAprt-realize of faeprograms and the programs from the archives without extraction, fSAprt-coding and fSAprt-translation may be used in high-intensity, ultra-short optical pulses laser of Nobel laureates 2018-year Gerard Mourou, Donna, Strickland. fSAprt– structure or an faeprogram if one is present of needed «target weight» are taken in target block at fSAprt– activation of the networks. $Q^{-1}(t)$ fSAprt Q derives fSAmnsprt to the sel$_Q$f -level boundary fSAmnsprt,g with target weight g.

It's used an alternating current of above high frequency and ultra-violet light, which can work with fSAprt– structures in fSAprt –modes by its nature to activate the networks or some of its parts in fSAprt –modes and locally using fSAprt –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."

3.18 FUZZY PROGRAM OPERATORS fSAprt, ftprSA, fS^{1}Aepr, fSAeprt$_1$ and their pself-type

Here it is supposed to use a symbiosis of parallel actions and conventional calculations through sequential actions [1]. This must be done through fSAprt-Networks - fuzzy analogue of Sit-Networks [15] in one of the central departments of which a conventional computer system is located. The parallel processor is itsel$_Q$f faeprogram - fuzzy analogue of eprogram [15] with direct parallel computing not through serial computing.

Using conventional coding by a computer system, through a Target-block with a fuzzy fSAprt -program operator - $Q^{-1}(t)$ fSAprt Q with actions Q, it will

be possible to obtain the fuzzy execution with actions Q 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 fSAprt-coding and fSAprt-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 faprogram of operator $\underset{Q^{-1}(t)}{\overset{activation}{}}\ \text{fSAprt}\ \underset{activation}{\overset{Q}{\{\text{UHF AC} := H\}}}$ with measure of fuzziness μ, we $\{\text{UHF AC} := H\}$ simultaneously enter the desired set of codes H 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 aprogram operators:

1) fSAprt-program operators
2) ftprSA-program operators
3) fS^{1}Aepr - program operators
4) fSAeprt$_{1}$- program operators

One example is pattern recognition: $\text{fSAprt}\ \underset{B}{\overset{image\ archive}{}}$ if $\text{fSAprt}\ \underset{B}{\overset{Q}{\ni}}\ \text{fSAprt}\ \underset{B}{\overset{q}{Q}}$ *then Name of q* .

The example of fSAprt-program is $\{\ \text{fSAprt}\ \underset{x}{\overset{Q}{\{\{p\} := \{\mathsf{a}(x)\}\}}}\ ,\ \text{fSAprt}\ \underset{x}{\overset{Q}{IF\{\{B\}\{f\}\}}}\ \underset{x}{\overset{Q}{then\ Q}}\ ,\ \text{fSAprt}\ \underset{Q}{\overset{Q}{Q}}\ \}$.

Consider a third type of fuzzy capacity in itsel$_Q$f - fuzzy analogue of capacity in itsel$_Q$f. For example, based on $\underset{q}{\overset{\tilde{x}}{\text{fSAprt}Q}}$, where $\tilde{x} = (x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \ldots, x_n|\mu_{\tilde{x}}(x_n))$, i.e. n - elements at one point, we can consider the capacity $fS_3Af\tilde{x}Q$ (in itsel$_Q$f with m elements from $\tilde{x}$, m<n, which is formed according to pQ-analogue of form (1.1), that is, the structure $\underset{q}{\overset{\tilde{x}}{\text{fSAprt}Q}}$ contains with actions Q only m elements or for more complex operators in p-analogue of forms (1.1.1) - (1.4), (2.1*) and analogues of forms (1.1.1) - (1.4) by type (2.1*).

Here are some of the fuzzy fSAprt-program operators.

The ideology of fSAprt and fS_3AfQ can be used for programming. Here are some of the fSAprt programming operators:

1. Simultaneous program operations Q with the expressions $\tilde{p} = (p_1|\mu_{\tilde{p}}(p_1), p_2|\mu_{\tilde{p}}(p_2)|,\ldots,p_n|\mu_{\tilde{p}}(p_n))$. This is implemented via $\underset{q}{\overset{\tilde{p}}{Q^{-1}\text{fSAprt}Q}}\overset{\tilde{p}}{}$.
2. Simultaneous program operations Q with pQ-checking the fuzzy set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2)|,\ldots,g_n|\mu_{\tilde{g}}(g_n)|)$ for the fuzzy set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2)|,\ldots,x_n|\mu_{\tilde{B}}(B_n)|)$ Implemented via $\underset{q}{\overset{q}{Q^{-1}\text{fSAprt}Q}}$. where $\tilde{H}$ can be anything.
3. Similarly for loop operators and others.

fS_3AfQ – fuzzy software operators will differ only in that the aggregates $\{\tilde{x}\},\{\tilde{p}\},\{\tilde{B}\},\{\tilde{g}\}$ will be formed from the corresponding fSAprt program operators in p-analogue of form (1.1) or forms (1.1.1) - (1.4), (2.1*) and analogues of p-analogue 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.

For example, $\underset{g\{R\}}{\overset{\{R\}}{\underset{Q}{\text{fSAprt}}}}$ is the fuzzy capacity with actions Q in itsel$_Q$f of the second type if $g\{R\}$ is a faprogram capable of fuzzy generating $\{R\}$ with actions Q.

The example of sel$_Q$f -faprogram of the first type is

$$\underset{\text{fSAprt}}{\{\underset{Q}{\underset{w}{\text{fSAprt}}}\{\{p\} := \{\mathsf{a}(x)\}\}, \underset{Q}{\underset{w}{\underset{Q}{\underset{w}{\text{fSAprt}}}}} IF\{\{B\}\{f\}\} \text{ then } Q, \underset{Q}{\text{fSAprt}}Q\}}.$$

The example of fSAprt-program for fSAmnsprt:

$\underset{q :=}{\overset{B}{\underset{Q^{-1}}{\text{fSAprt}}}}\underset{B}{\overset{q :=}{Q}}$ - fuzzy p-assigning fuzzy q to fuzzy B with actions Q.

$\underset{tw :=}{\overset{g}{\underset{Q^{-1}}{\text{fSAprt}}}}\underset{g}{\overset{tw :=}{Q}}$ - fuzzy p-assigning target weight *tw* to fuzzy g with actions Q.

$\underset{\{q\}w}{\overset{\{q\}w}{\underset{Q^{-1}}{\text{fSAprt}}}}\underset{f\text{SAmnsprt } activation}{\overset{f\text{SAmnsprt } activation}{Q}}$ - fSAmnsprt p-*activation* for fuzzy $\{q\}w$ with actions Q.

fSAprt-coding.

fSAprt-coding with measure of fuzziness Q: 1) fuzzy set A to fuzzy set B, fuzzy set D from fuzzy set C, 2) fuzzy set A to a point q, where the elements of the fuzzy sets A, B can be continuous. For example, $\underset{D}{\overset{C}{\underset{Q^{-1}}{\text{fSAprt}}}}\underset{B}{\overset{A}{Q}}$.

There are fSAprt-coding, fSAprt-translation, fSAprt-realize of faeprograms and of the faprograms from the archives without extraction theirs

sel$_Q$f-coding.

sel_Qf -coding with actions Q: 1) fuzzy set A to set fuzzy A, i.e. fuzzy A on it sel_Qf 2) fuzzy set A to a point q in p-analogue of form (1.1), where the elements of the fuzzy sets A, B can be continuous. For example, ${}^{A}_{A}Q^{-1}\text{fSAprt}^{A}_{A}Q$ or ${}^{B}_{A}Q^{-1}\text{fSAprt}^{A}_{B}Q$ etc.

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 fSAprt-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 fSAprt-Networks and series-parallel. Codes from a conventional type computer system will be used via fSAprt -connectors in fSAprt - coding, for example: ${}^{\tilde{p}}_{q}Q^{-1}\text{fSAprt}$

$\{\text{UHF AC}\underset{activation}{:=}\underset{Q}{}\}$. UHF AC field activation is used.

Dynamic fSAprt and fS_3AfQ programming [1].

The ideology of dynamic fSAprt and fS_3AfQ can be used for programming: DynamicfSAprt and $fS_iAf_{Q(t)}$ programming [1].

The ideology of dynamicfSAprt and $fS_3Af_{Q(t)}$can be used for programming:

1. The process of simultaneous program actions Q(t) of the fuzzy expressions $\widetilde{p(t)}=(p_1(t)|\mu_{\widetilde{p(t)}}(p_1(t)), p_2(t)|\mu_{\widetilde{p(t)}}(p_2(t))|_{,...,}p_n(t)|\mu_{\widetilde{p(t)}}(p_n(t)))$ to the fuzzy variables $\widetilde{x(t)}=(x_1(t)|\mu_{\widetilde{x(t)}}(x_1(t)), x_2(t)|\mu_{\tilde{x}(t)}(x_2(t))|_{,...,}x_n(t)|$ $\mu_{\widetilde{x(t)}}(x_n(t)))$. This is implemented via ${}^{\widetilde{w(t)}}_{\widetilde{p(t)}}Q^{-1}{}_{(t)}\text{fSAprt}(t)^{\widetilde{p(t)}}_{\widetilde{w(t)}}Q_{(t)}$.
2. The process of simultaneous pQ-checking with program actions Q(t) 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 $\widetilde{B(t)}=(B_1(t)|$ $\mu_{\widetilde{B(t)}}(B_1(t)), B_2(t)|\mu_{\tilde{B}(t)}(B_2(t))|_{,...,}B_n(t)|\mu_{\widetilde{B(t)}}(B_n(t)))$. Implemented via

$\overset{w(\widetilde{t})}{Q^{-1}(t)}$ fSAprt(t) $\overset{IF\left\{\left\{B(\widetilde{t})\right\}\left\{\tilde{g}(\widetilde{t})\right\}\right\}\ then\ H(\widetilde{t})}{Q(t)}$ where $IF\left\{\left\{B(\widetilde{t})\right\}\left\{\tilde{g}(t)\right\}\right\}\ then\ H(\widetilde{t})$ $\overset{}{w(\widetilde{t})}$

$H(\widetilde{t})$ can be anything.

3. Similarly for fuzzy loop operators and others.

$fS_1Af_{Q(t)}$- fuzzy fsoftware operators will differ only just because aggregates $x(\widetilde{t}),p(\widetilde{t}), B(\widetilde{t}),g(\widetilde{t})$ will be formed from corresponding processes by fSAprt-program operators in p-analogue of form (1.1) for more complex operators in p-analogue of forms (1.1.1) - (1.4), (2.1*) and analogues of forms (1.1.1) - (1.4) by type (2.1*).

ftprSA-program operators.

The ideology of ftprSA and ft_{S_4Af} - fuzzy analogues of tS and t_{S_4f} from [10] can be used for programming. Here are some of the ftprSA -program operators.

4. Simultaneous expelling assignment of the expressions $\tilde{p}=(p_1|\mu_{\tilde{p}}(p_1), p_2|\mu_{\tilde{p}}(p_2), \ldots, p_n|\mu_{\tilde{p}}(p_n))$ from the variables $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \ldots, x_n|\mu_{\tilde{x}}(x_n))$ with actions Q. This is implemented via $\overset{\tilde{x}}{Q}$ fSAprt. $=:\tilde{p}$
5. Simultaneous expelling check with actions Q the fuzzy set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2), \ldots, g_n|\mu_{\tilde{g}}(g_n))$ for the fuzzy set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2), \ldots, B_n|\mu_{\tilde{B}}(B_n))$. It's implemented through $\overset{w}{Q}$ $IF\left\{\left\{\tilde{B}\right\}\left\{\tilde{g}\right\}\right\}\ then\ \tilde{Q}$ fSAprt,
6. where can Q be any.
7. Similarly for loop operators and others.

ft_{S_4Af} – fuzzy software operators will differ only just because aggregates $\tilde{x},\tilde{p}, \tilde{B},\tilde{g}$ will be formed from corresponding ftprSA program operators in os-analogue of form (1.1) for more complex operators in os-analogue of forms (1.1.1) - (1.4),

(2.1*) and analogues of forms (1.1.1) - (1.4) by type (2.1*). Consider hierarchical ftprSA-program operator

$$\overset{B}{Q}\text{fSAprt} = \left\{ \begin{matrix} D + \overset{\{\}}{Q}\ \text{fSAprt} \\ A - A \cap B \\ (B - A \cap B) \end{matrix} \right\}, \text{ where D is osel}_Q\text{f-(fuzzy set) for fuzzy } (A \cap B).$$

Dynamic ftprSA and $ft(q)_{S_4Af}$ programming at time q.

The ideology of ftprSA and ft_{S_4Af} can be used for dynamic programming. Here are some of the ftprSA()- 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 $\widetilde{x(t)}$=($x_1(t)|\mu_{\widetilde{x(t)}}(x_1(t))$, $x_2(t)|\mu_{\tilde{x}(t)}(x_2(t))$, ..., $x_n(t)|\mu_{\widetilde{x(t)}}(x_n(t))$) with actions Q is implemented through

$$\overset{\tilde{x}(t)}{Q}\ \text{fSAprt(t).}$$
$$= :\tilde{p}(t)$$

2. The process of simultaneous expelling check with actions Q the fuzzy set of conditions $\widetilde{g(t)}$=($g_1(t)|\mu_{\widetilde{g(t)}}(g_1(t))$, $g_2(t)|\mu_{\tilde{g}(t)}(g_2(t))$, ..., $g_n(t)|\mu_{\widetilde{g(t)}}(g_n(t))$) for the fuzzy set of expressions $\widetilde{B(t)}$=($B_1(t)|\mu_{\widetilde{B(t)}}(B_1(t))$, $B_2(t)|\mu_{\tilde{B}(t)}(B_2(t))$, ..., $B_n(t)|\mu_{\widetilde{B(t)}}(B_n(t))$) is implemented through $\overset{w(t)}{Q}\ \text{fSAprt(t)}$, 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.

ft_{S_4Af} – 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 ftprSA(t) for above mentioned programming operators through os-analogue of form (1.1) for more complex operators in os-analogue of forms (1.1.1) - (1.4), (2.1*) and analogues of forms (1.1.1) - (1.4) by type (2.1*). Consider hierarchical dynamic ftprSA-program operator:

$$\underset{A(q)}{\overset{B(q)}{Q}}\ \text{fSAprt(q)} = \left\{ ft(q)_{S_1 Af(A(q)\cap B(q))} + \underset{A(q)-A(q)\cap B(q)}{\overset{\{\}}{Q}}\ \text{fSAprt(q)} \right\}, \\ (B(q)-A(q)\cap B(q))$$

fS^{1}Aepr -program operators (form $\underset{D}{\overset{B}{Q_2}}\text{ffS}^1\text{prt}\underset{B}{\overset{A}{Q_1}}$ - fuzzy analogue of ${}_D^B S^1 t_B^A$ [6])**)**

For example, $\underset{\{=:\{p(t)\}\}}{\overset{\{\mathsf{a}(t)\}}{Q_2}}\ \text{fS}^1\text{Aeprt}\ \underset{\{\mathsf{a}(t)\}}{Q_1}$ $IF\{\{B(t)\}\{f(t)\}\}\ then\ Q(t)$.

Consider hierarchical dynamic fS^{1}Aepr -program operator: (form

$$\underset{A}{\overset{B}{Q_2}}\text{fS}^1\text{Aprt}\underset{B}{\overset{A}{Q_1}} * \underset{D-A}{\overset{B}{Q_2}}\ \text{fS}^1\text{Aprt}\underset{B}{\overset{A}{Q_1}}\ ).$$

fSAeprt$_1$- program operators (form $\underset{D}{\overset{C}{Q_2}}\text{ffS}_1\text{prt}\underset{B}{\overset{A}{Q_1}}$ - fuzzy analogue of ${}_D^C S_1 t_B^A$ [11])**)**

Consider examples

1. $\text{fSAprt}\left\{ \underset{W_q}{\overset{q}{\left(\begin{smallmatrix} a & a \\ Q_1\text{fSArt}Q_1 & \\ a & a \end{smallmatrix}\right)}} fSprt_{q\left(\begin{smallmatrix} a & a \\ Q_1\text{fSArt}Q_1 & \\ a & a \end{smallmatrix}\right)}^{E_q},\quad fSprt_{d_r\left(E_{in}{}^{l^{d_r}}\right)}^{\{E^{ex}{}_l{}^{d_r}\}} \right\}$ —faprogram structure

$\underset{t_0}{\overset{Q}{\ }}$

example, where the assemblage point d_r is the cursor, it is quite complex sel_Qf —faprogram.

2. $\text{fSAprt}\left\{ \underset{W_q}{\overset{q}{\left(\begin{smallmatrix} a & a \\ Q_1\text{fSArt}Q_1 & \\ a & a \end{smallmatrix}\right)}} fSprt_{q\left(\begin{smallmatrix} a & a \\ Q_1\text{fSArt}Q_1 & \\ a & a \end{smallmatrix}\right)}^{E_q},\quad fSprt_{d_r\left(E_{in}{}^{l^{d_r}}\right)}^{\{E^{ex}{}_l{}^{d_r}\}} \right\}$ can be interpreted as a

$\underset{t_0}{\overset{Q}{\ }}$

faprogram operator.

$\overset{a}{\underset{a}{Q_1}}$fSArt$\overset{a}{\underset{a}{Q_1}}$ can be interpreted as a $\begin{pmatrix} s_1elf \\ os_1elf \end{pmatrix}$-faprogram operator. $\overset{a}{\underset{a}{Q_1}}$fSArt$\overset{a}{\underset{a}{Q_1}}$--sample $\begin{pmatrix} s_1elf \\ os_1elf \end{pmatrix}$-faprogram structure example.

Appendix

Remark. Energy of a living organism:

$$\text{afg(r, a}(E_q)) = \text{fSAprt}\left\{ \begin{matrix} {}^{q}_{\left(\begin{smallmatrix} a & a \\ Q_1\text{fSArt}Q_1 \\ a & a \end{smallmatrix}\right)} W_q fSprt^{E_q}_{q_{\left(\begin{smallmatrix} a & a \\ Q_1\text{fSArt}Q_1 \\ a & a \end{smallmatrix}\right)}}, & fSprt^{\{E^{ex}l^{d_r}\}}_{d_r\left(E_{in}l^{d_r}\right)} \end{matrix} \right\}_{\substack{Q \\ t_0}} (**_{3.7}).$$

$\overset{a}{\underset{a}{Q_1}}$fSArt$\overset{a}{\underset{a}{Q_1}}$-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 sel_Qf -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 sel_Qf -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 actions Q_1, Q of a living organism of a person:

$$\text{afpg(r, a}(E_q)) = \left\{ \begin{matrix} {}^{q}_{\left(\begin{smallmatrix} a & a \\ Q_1\text{fSArt}Q_1 \\ a & a \end{smallmatrix}\right)} W_q fSprt^{E_q}_{q_{\left(\begin{smallmatrix} a & a \\ Q_1\text{fSArt}Q_1 \\ a & a \end{smallmatrix}\right)}}, & fSprt^{\{E^{ex}l^{d_r}\}}_{d_r\left(self\left(E_{in}l^{d_r}\right)\right)} \end{matrix} \right\}_{\substack{Q \\ t_0}} (***_{3.7}).$$

$(**_{3.7})$, $(***_{3.7})$ can be interpreted as fSArt - program operators.

Entire neural network as instantaneous simultaneous afRAM in fSAprt-elements and sel_Qf - elements. $fsel_Qf^{fsel_Qf^{\cdots^{fsel_Qf}}}$, $\text{ff1}\downarrow \text{I} \uparrow_{-1}^{1} ff_2$ $^{\text{ff1}\downarrow \text{I} \uparrow_{-1}^{1} ff_2 \cdots^{\text{ff1}\downarrow \text{I} \uparrow_{-1}^{1} ff_2}}$,

$fsin\infty^{fsin\infty^{\cdots^{fsin\infty}}}$.When activated in a neural network, the entire neural network becomes a working memory. Use of $sel_Q f$ -energy as fuzzy activation or from outside. afQ_0= $\overset{fSA_{mnSprt}}{\underset{Q}{\overset{\text{fSAprt}}{\underset{activation}{}}}}$ → $sel_Q f$ -afRAM, afQ_{00}= $\overset{fSA_{mnSprt}}{\underset{Q}{\overset{\text{fSAprt}}{\underset{activation}{}}}}$ $\overset{fSA_{mnSprt}}{\underset{Q}{\overset{\text{fSAprt}}{\underset{activation}{}}}}$ afQ_0prt, afQ_{01}= $\overset{fSA_{mnSprt}}{\underset{Q}{\overset{\text{fSAprt}}{\underset{activation}{}}}}$ afQ_0.afQ_0,afQ_{00},afQ_{01}- $\overset{fSA_{mnSprt}}{\underset{Q}{\overset{\text{fSAprt}}{\underset{activation}{}}}}$ coding,translation,realization in faeprograms, use afQ_0,afQ_{00},afQ_{01}- fSA_{mnSprt},afAssembler.

3.19 Introduction to FUZZY PROGRAM OPERATORS Flprt, tprFL, FL^1epr, $FLeprt_1$

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 fdeprogram - 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 fuzzy Lprt -program operator - $\overset{activation\ g}{\underset{\overset{\sim\sim}{A}}{\overset{\uparrow}{\underset{\uparrow}{\overset{\overline{P}}{\underset{\uparrow}{}}}}}}$ FLprt $\overset{\overset{\sim\sim}{A}}{\underset{activation\ g}{\overset{\uparrow}{\underset{\uparrow}{\overset{\overline{Q}}{\underset{\uparrow}{}}}}}}$, 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 $\mu_{activation}$ and performs the inverse action of P with measure of fuzziness μ_P from fuzzy *activation* with measure of fuzziness $\mu_{activation}$ simultaneously, Q, P are any fuzzy *action*s, 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 operator

$$\begin{matrix} activation \\ action\ Q^{-1} \\ \{\text{UHF AC} := R\} \end{matrix} \quad \text{f}\overleftarrow{Dprt} \quad \{\text{UHF AC} := R\} \begin{matrix} \\ action\ Q \\ activation \end{matrix}$$

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 Lprt-program operators (designation FLprt-program operators)
3. fuzzy tprL-program operators (designation tprFL-program operators)
4. fuzzy L^1epr - program operators (designation FL^1epr -program operators)
5. fuzzy $Leprt_1$- program operators (designation $FLeprt_1$-program operators)

Examples:

$$
\begin{array}{lll}
 & & \overset{\leftrightarrow}{\overset{\leftrightarrow}{A}} \\
 & & \uparrow \\
1)\ \text{Lprt} & \textit{then}\ \overline{Q} & \\
 & \uparrow & \\
 & IF\{\{B\}\{f\}\} & \\
C & & \overset{\leftrightarrow}{\overset{\leftrightarrow}{A}} \\
\uparrow & & \uparrow \\
2)\ \overline{P} & \text{Lprt}\ \textit{then}\ \overline{Q}: \equiv & \\
\uparrow & & \uparrow \\
\overset{\leftrightarrow}{\overset{\leftrightarrow}{R}} & IF\{\{B\}\{f\}\} & \\
C & & \overset{\leftrightarrow}{\overset{\leftrightarrow}{A}} \\
\uparrow & & \uparrow \\
3)\ \overline{P} & \text{Lprt}\ \overline{Q}: \equiv & \\
\uparrow & & \uparrow \\
\overset{\leftrightarrow}{\overset{\leftrightarrow}{R}} & B := &
\end{array}
$$

etc.

3.20 Introduction to FUZZY PROGRAM OPERATORS FWprt, tprFW

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], [15-20] 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], [15-20] with direct parallel computing not through serial computing.

Using conventional coding by a computer system, through a Target-block with a fuzzy Wprt -program operator - FWprt $\begin{matrix}\tilde{\tilde{\tilde{H}}}\\ \uparrow\\ \overline{\overline{F}}\\ \uparrow\\ \tilde{\tilde{A}}\\ \uparrow\\ \overline{Q}\\ \uparrow\\ activation\ g\end{matrix}$, where fuzzy H 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 H 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 FWprt-coding and FWprt-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 fwdprogram of operator fDprt

{UHF AC $:= R$}

$\begin{matrix}action\ Q\\ activation\end{matrix}$ 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 with minimal amplitude and maximum frequency 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 fwdprogram operators:

1) fuzzy Lprt-program operators (designation FWprt-program operators)
2) fuzzy tprW-program operators (designation tprFW-program operators)
3) fuzzy W[1]epr - program operators (designation FW[1]epr -program operators)
4) fuzzy Weprt$_1$- program operators (designation FWeprt$_1$-program operators)

Examples:

1) FWprt

$$\begin{array}{c} \overset{\cdots}{\overbrace{H}} \\ \uparrow \\ \overline{\overline{F}} \\ \uparrow \\ \overbrace{A} \\ \uparrow \\ \overline{then\ \overline{Q}} \\ \uparrow \\ IF\{\{B\}\{f\}\} \end{array}$$

2) FWprt

$$\begin{array}{c} C \\ \uparrow \\ \overline{P} \\ \uparrow \\ \overbrace{R} \\ \uparrow \\ \overline{\overline{G}} \\ \uparrow \\ \overbrace{O} \\ \cdots \end{array} \quad \begin{array}{c} \overset{\cdots}{\overbrace{H}} \\ \uparrow \\ \overline{\overline{F}} \\ \uparrow \\ \overbrace{A} \\ \uparrow \\ \overline{then\ \overline{Q}: \equiv} \\ \uparrow \\ IF\{\{B\}\{f\}\} \end{array}$$

3) FWprt

$$\begin{array}{c} C \\ \uparrow \\ \overline{P} \\ \uparrow \\ \overbrace{R} \\ \uparrow \\ \overline{\overline{G}} \\ \uparrow \\ \overbrace{O} \\ \cdots \end{array} \quad \begin{array}{c} \overset{\cdots}{\overbrace{H}} \\ \uparrow \\ \overline{\overline{F}} \\ \uparrow \\ \overbrace{A} \\ \uparrow \\ \overline{Q}: \equiv \\ \uparrow \\ B := \end{array}$$

3.21 Introduction to FUZZY PROGRAM OPERATORS FZprt, tprFZ

Here it is supposed to use a symbiosis of parallel actions and conventional calculations through sequential actions. This must be done through FZprt-Networks - fuzzy analogue of Sit-Networks [1], [6], [12], [15-20] 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], [15-20] with direct parallel computing not through serial computing.

Using conventional coding by a computer system, through a Target-block with a fuzzy Zprt -program operator - FZprt $\begin{array}{c}\overset{\cdots}{\overset{\sim}{\overset{\sim}{\tilde{H}}}}\\ \uparrow \\ \overline{\overline{F}} \\ \uparrow \\ \tilde{A} \\ \uparrow \\ \overline{Q} \\ \uparrow \\ activation\ g\end{array}$, where fuzzy H 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 H 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 FZprt-coding and FZprt-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 fwdprogram of operator fDprt

$\{\text{UHF AC} := R\}$
$\begin{matrix} action\ Q \\ activation \end{matrix}$ 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 with minimal amplitude and maximum frequency 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 fwdprogram operators:

1) fuzzy Zprt-program operators (designation FZprt-program operators)

2) fuzzy tprZ-program operators (designation tprFZ-program operators)

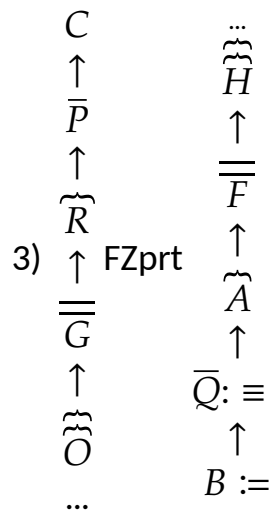
Examples:

1) FZprt

$$\begin{array}{c} \overset{\cdots}{\overbrace{H}} \\ \uparrow \\ \overline{\overline{F}} \\ \uparrow \\ \overbrace{A} \\ \uparrow \\ \overline{then\ \overline{Q}} \\ \uparrow \\ IF\{\{B\}\{f\}\} \end{array}$$

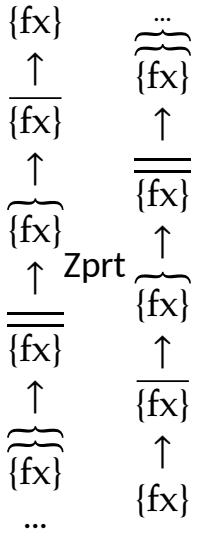
2) FZprt

$$\begin{array}{c} C \\ \uparrow \\ \overline{P} \\ \uparrow \\ \overbrace{R} \\ \uparrow \\ \overline{\overline{G}} \\ \uparrow \\ \overbrace{O} \\ \cdots \end{array} \qquad \begin{array}{c} \overset{\cdots}{\overbrace{H}} \\ \uparrow \\ \overline{\overline{F}} \\ \uparrow \\ \overbrace{A} \\ \uparrow \\ \overline{then\ \overline{Q}: \equiv} \\ \uparrow \\ IF\{\{B\}\{f\}\} \end{array}$$

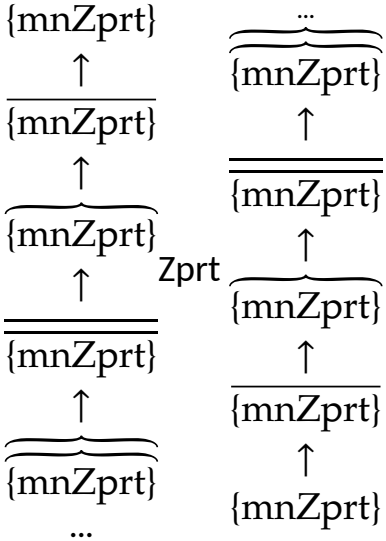


3.22 Zprt-networks

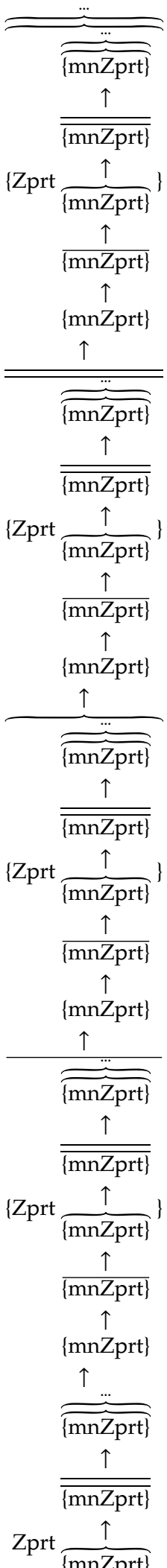
A. Galushkin's comprehensive monograph [21] covers all aspects of networks, but traditional approaches go through classical mathematics, mainly through the usual correspondence operators. Here we consider a different approach - through a new mathematical process with containment operators, which, although 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 Zprt-networks. Zprt networks (SmnZprt) are a Zprt 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. Zprt-OS (Zprt operating system) uses Zprt-coding and Zprt-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. Zprt-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. Zprt-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) [22] will be more stable. The set of the

target weights $f = (f_1, f_2, \ldots, f_n)$ in  are chosen for necessary

tasks using a short-pulse laser to generate attosecond pulses of light to accomplish them, $x= (x_{1,} x_{2,} \ldots, x_n)$. We will not touch on the issues of applications, or network optimization. They are described in detail by A. Galushkin [21]. 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 Zprt (designation - mnZprt) for simple information processing. More complex executing programs are used for mnZprt nodes. The first level of mnZprt consists of simple mnZprt. The

second level of mnZprt consists of  – Zprt-node of

mnZprt. The third level of mnZprt consists of Zprt



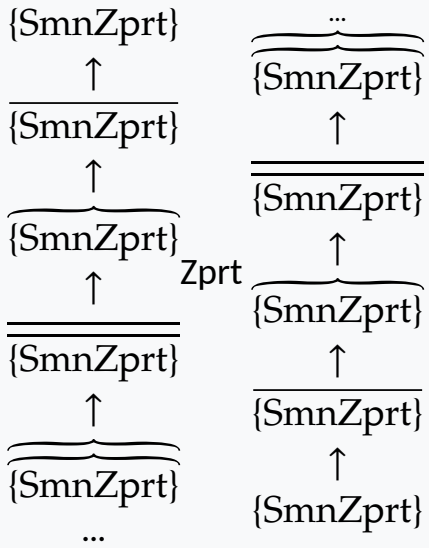
- $Zprt^2$- node of mnZprt. The target structure or the corresponding program enters the target unit using a short-pulse laser to generate attosecond pulses of light. After that, all networks or parts of them are activated according to the indicative goal. It may appear that we are leaving the network ideology, but these networks are a complex hierarchy of different levels, like living organisms.

Remark 3.21. 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. mnZprt contains *w*eprograms –executing program in Zprt-OS. Zprt-OS (or Self-type of Zprt-OS) is based on Zprt-assembly language (or Self-type of Zprt-assembly language), which is based on assembly language through Zprt-approach in turn, if the base of elements of Zprt-networks is sufficient. The reprograms are in Zprt-programming environments (or Self-type of Zprt programming environments), but this question and Zprt-networks base will be considered in the following articles. In particular, weprograms may contain Zprt- programming operators. In mnZprt cores, the constant memory Zprt with correspondent weprograms depending on mnZprt.

The OS (operating system) and the principles and modes of operation of the Zprt-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 Zprt – physics and special neural networks with artificial neurons operating in normal and Zprt-modes. Let's denote this model through SmnZprt. To do this, it’s proposed to use mnZprt of different levels; for example, for the usual mode, mnZprt 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 Zprt–mode with the desired "target weight" is realized in this section, etc, to the center. In the case of a monster

during the test, SmnZprt 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



is used. In normal mode, it's planned to carry out the movement of SmnZprt 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 SmnZprt 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 SmnZprt. SmnZprt is represented by a neural network that extends from the center of one of the main clusters of Zprt - 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 SmnZprt's actions is below the operator's cab. In Zprt – mode, the entire network or its sections are Zprt – 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 Zprt -coding, Zprt-translation for activation of all networks to "target weights" simultaneously, then –the reset of this Zprt-coding after activation using a short-pulse laser to generate attosecond pulses of light.

Unfortunately, triodes are not suitable for Zprt -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 Zprt-operative memory belt is disposed around a central core of SmnZprt. There are Zprt-coding, Zprt-translation, and Zprt-realize of weprograms and the programs from the archives without extraction, Zprt-coding and Zprt-translation may be used in high-intensity, ultra-short optical pulses laser of Nobel laureates 2018-year Gerard Mourou, Donna, Strickland. Zprt – structure or an reprogram if one is present of needed «target weight» are taken in target block at Zprt – activation of the networks.

$$\begin{array}{c} \{\text{SmnZprt, f}\} \\ \uparrow \\ \overline{\{\text{SmnZprt, f}\}} \\ \uparrow \\ \overbrace{\{\text{SmnZprt, f}\}} \\ \uparrow \\ \overline{\overline{\{\text{SmnZprt, f}\}}} \\ \uparrow \\ \overbrace{\overbrace{\{\text{SmnZprt, f}\}}} \\ \ldots \end{array} \text{Zprt} \begin{array}{c} \overbrace{\overbrace{\ldots}} \\ \overbrace{\{\text{SmnZprt, f}\}} \\ \uparrow \\ \overline{\overline{\{\text{SmnZprt, f}\}}} \\ \uparrow \\ \overbrace{\{\text{SmnZprt, f}\}} \\ \uparrow \\ \overline{\{\text{SmnZprt, f}\}} \\ \uparrow \\ \{\text{SmnZprt, f}\} \end{array}$$

derives SmnZprt to the 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, which can work with Zprt– structures in Zprt–modes by its nature to activate the networks or some of its parts in Zprt–modes and locally using Zprt–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 $(a \uparrow \mathrm{I} \downarrow^{a}) \uparrow$

$\mathrm{I}\downarrow(\downarrow_a \downarrow \mathrm{I}^{\uparrow a})$, 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 [15-20] SmnZprt . Execution tw will be achieved through setting the assemblage point in the desired position r_1 to engage the appropriate external energy: $Sprt^{d_r}_{d_r}{}^{r}Sprt^{d_r}_{\mathrm{r1}}$ [15-20].

3.23 Introduction to FUZZY PROGRAM OPERATORS $\mathrm{FV_1prt}$

The program operator corresponds to vector of energy levels of a non-living object, A is program element []:

$$\mathrm{B(A)}=\begin{pmatrix} \overset{\cdots}{} \\ parel\,fA\left(decignation-\overline{\overline{\overline{A}}}\right) \\ singel\,fA\left(decignation-\overline{\overline{A}}\right) \\ subtle\ energy\ of\ object\ \ A\ \text{paradoxical}\ upper\ level\ (\ pa|||)\left(decignation-\overline{\overline{\overline{A}}}\right) \\ subtle\ energy\ of\ object\ \ A\ \text{paradoxical}\ mid-level\left(decignation-\overline{\overline{\overline{A}}}\right) \\ \begin{matrix} / \\ subtle\ energy\ of\ |||^{-1} \\ osel\,f(A) \end{matrix} \qquad\qquad \begin{matrix} \backslash \\ subtle\ energy\ of\ ||| \\ \widehat{A} \\ \left(\overline{A}\right) \end{matrix} \\ ordinary\ energy\ exhibited\ by\ an\ \ object\ \ A\left(decignation-\underline{A}\right) \ \leftarrow\ \ the\ raw\ energy\ of\ an\ object\ \ A\left(decignation-A\right) \end{pmatrix},$$

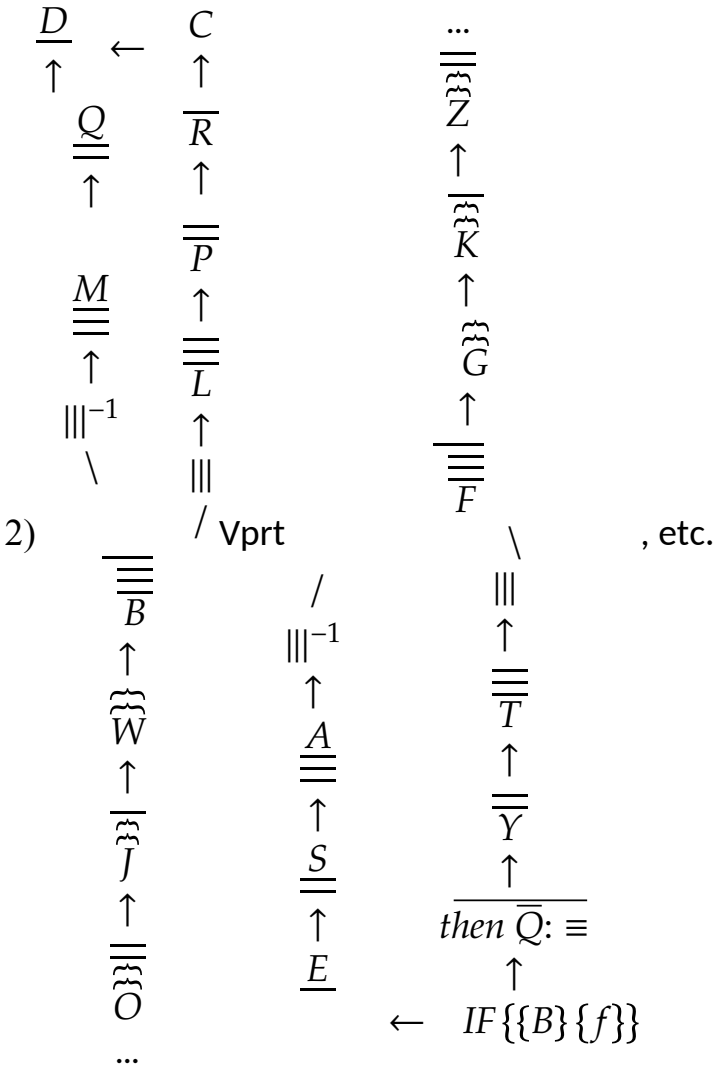
$\overline{A}=self(A)$, $\widehat{A}=self^{\frac{3}{2}}(containment\ for\ creating\ of\ A)$.

The program operator corresponds to vector of energy levels of a living object, A is program element []:

$$\mathrm{C(A)} = \left(\begin{array}{c} \cdots \\ parel f A\left(decignation - \overline{\overline{\overline{A}}} \right) \\ singel f A\left(decignation - \overline{\overline{\overline{A}}} \right) \\ subtle\ energy\ of\ object\ \ A\ \text{paradoxical}\ upper\ level\ (\ pa|||)\left(decignation - \overline{\overline{\overline{A}}} \right) \\ subtle\ energy\ of\ object\ \ \ A\ \text{paradoxical}\ mid - level(pasel f(A))\left(decignation - \overline{\overline{\overline{\overline{A}}}} \right) \\ subtle\ energy\ of\ object\ \ \ A\ \text{paradoxical}\ down - level(psel f(A))\left(decignation - \overline{\overline{\overline{A}}} \right) \\ \begin{array}{cc} / & \backslash \\ subtle\ energy\ of\ |||^{-1} & subtle\ energy\ of\ ||| \\ \underline{\underline{\underline{A}}} & \overline{\overline{\overline{A}}} \\ \left(\underline{\underline{\underline{A}}}\right) & \left(\overline{\overline{\overline{A}}}\right) \end{array} \\ ordinary\ energy\ exhibited\ by\ an\ \ object\ \ A\left(decignation - \underline{\underline{\underline{A}}} \right) \leftarrow \ the\ raw\ energy\ of\ an\ object\ \ A\left(decignation - A \right) \end{array} \right)$$

$\overline{\overline{\overline{A}}} = a$, $\overline{\overline{\overline{A}}} = \mathrm{Q^{*}self(P|||)}$.

May consider the program operators: 1) $\mathrm{C(A)VprtB^{-1}(A)}$,



3.24 Interpretations forms programming by containment

Any interpretation form may be used for programming. For example, (1, (2, 1))-program: self($Sprt_{x}^{IF\{\{B\}\{f\}\}\ then\ Q}$), (1, (2, 1))-programming is programming itself, i.e., it can reprogram itself and be in constant development and can program all inside itself. In time slices, ready-made software environments can be obtained from here.

A-programming, A is any, in particular, any interpretation.

$D_{(A, (B, C))}$-programming, not to be confused with programming, where $D_{|A} := D_{|(B, C)}$.

(2, 1)-programming is ||| of all, whatever it needs, including itself with whatever it needs.

(1, 2) -programming is $|||^{-1}$ of all, whatever it needs, including itself.

To use the program operator $\overset{task\ B}{solution\ of\ B}$Sprt by SmnSprt need train it, in particular, through relevant examples. Sprt$\overset{f(B)}{B}$-programming is transformation of B-programming to $f(B)$ -programming, in particular, with exit to upper level.

The program operator $\overset{B}{A}$Sprt$\overset{A}{B}$ allows made the field of oscillatory programming by containment. Next are $\overset{A}{A}$Sprt$\overset{A}{A}$-programming, A|||B-programming, A|||A-programming, paself(A)-programming, singelf(A)-programming, parelf(A)-programming etc. A, B are any, in particular, programs.

The program operator $\overset{B}{A}$Sprt$\overset{A}{B}$ creates an energy field for programming by containment. Example of connection to a field: $\overset{B}{A}$Sprt$\overset{C}{B}$. Example of rectifying variability: Sprt$\overset{A}{B}$.

The dynamic operators $\begin{matrix}\text{program } C\\ \text{program } D\end{matrix}$Sprt$\begin{matrix}\text{program } A\\ \text{program } B\text{'}\end{matrix}$ $\begin{matrix}\text{program } B\\ \text{program } A\end{matrix}$Sprt$\begin{matrix}\text{program } A\\ \text{program } B\text{'}\end{matrix}$ $\begin{matrix}\text{program } A\\ \text{program } A\end{matrix}$Sprt$\begin{matrix}\text{program } A\\ \text{program } B\text{'}\end{matrix}$ $\begin{matrix}\text{program } B\\ \text{program } A\end{matrix}$Sprt$\begin{matrix}\text{program } A\\ \text{program } A\text{'}\end{matrix}$ $\begin{matrix}\text{program } A\\ \text{program } A\end{matrix}$Sprt $\begin{matrix}\text{program } A\\ \text{program } A\text{'}\end{matrix}$ $\begin{matrix}\text{program } B\\ \text{program } A\end{matrix}$Sprt, $\begin{matrix}\text{program } A\\ \text{program } A\end{matrix}$Sprt Sprt$\begin{matrix}\text{program } A\\ \text{program } A\text{'}\end{matrix}$, Sprt$\begin{matrix}\text{program } A\\ \text{program } B\end{matrix}$ and others allow to work with programs.

3.25 Interpretations forms programming by other actions

Some variants for direct-objective programming, direct- parallel programming, direct-accumulative programming

Dprt-programming

The program operator $\overset{a}{\underset{b}{Q^{-1}}}\text{Dprt}\ \overset{b}{\underset{a}{action}}\ Q$ allows made the field of oscillatory programming by Q, in particular, Q = |||.

$\overset{a}{\underset{b}{Q^{-1}}}\overrightarrow{Dprt}\ \overset{b}{\underset{a}{action}}\ Q$, in particular, Q = |||.

The program operator $\overset{a}{\underset{b}{Q^{-1}}}\overrightarrow{Dprt}\ \overset{b}{\underset{a}{action}}\ Q$ allows made the ordered field of oscillatory programming by Q, in particular, Q = |||.

Rprt-programming

The program operator $\overset{a}{\underset{b}{P}}\text{Rprt}\ \overset{b}{\underset{a}{action}}\ Q$ allows made the field of oscillatory programming by Q, in particular, Q = |||, P =.

The program operator $\overset{a}{\underset{b}{P}}\text{R}\overrightarrow{prt}\ \overset{b}{\underset{a}{action}}\ Q$ allows made the ordered field of oscillatory programming by Q, in particular, Q = |||, P =.

The direct- parallel programs can have the following types:

a) “friends”, i.e., joint execution,
b) “rivals”,
c) Neutral variants
d) Etc.

In particular, the solution algorithm f(x) = 0, $\|f\| < 1$, can be replaced with x = $\lim\limits_{n \to \infty} f^n(x_0)$.

New programming with containment and partial assignment (replacement).

In particular, partial assignment (replacement) allows to give the new type of self: chself by partial assignment (replacement) of set A :=$_p$ itself, where :=$_p$ is the sign

of partial assignment (replacement), for example, set A replaces by itself as an element of itself. May consider the partial assignment (replacement) of set A and inverse operations simultaneously as pchself, pachself = (A :≡$_p$ (-A)), pachrelf, chsingelfg, self-program operator of cycle FORitself, self-program operator of checking IFitself etc. The operators :≡$_p$, :≡, (:≡ itself as element) are operators, which create singularities.

3.26 Nth-programming

Definition. Biself(A, B) is a program A contains into program B as element and B contains into A as element simultaneously: $\begin{matrix}\text{Sprt}^{A}_{B}\\ \text{Sprt}^{B}_{A}\end{matrix}$.

Definition. Nthself(A_1, A_2, …, A_N) is program A_j contains program A_i as element, ∀j, i = 1, 2, …, N simultaneously.

Definition. STRself(A) is any part Q of program A contains any other part of A as element simultaneously.

Definition. STpRself(A) is any part Q of program A contains any other part of A as element partially and simultaneously.

Definition. STp$_1$Rself(A) is any part Q of program A partially contains any other part of A as element simultaneously.

Etc.

Definition. oBiself(A, B) is program A expelling from program B as element and B expelling from A as element and the first from the second simultaneously: $\begin{matrix}\text{Sprt}^{B}_{A}\\ \text{Sprt}^{B}_{A}\end{matrix}$

Sprt.

Definition. oNthself(A_1, A_2, …, A_N) is program A_j contains and expelling from program A_i as element, ∀j, i = 1, 2, …, N simultaneously.

Definition. oSTRself(A) is any part Q of program A contains and expelling from any other part of A as element simultaneously.

Definition. oSTpRself(A) is any part Q of program A contains and expelling from any other part of A as element partially and simultaneously.

Definition. oSTp$_1$Rself(A) is any part Q of program A partially contains and partially expelling from any other part of A as element simultaneously.

Etc.

Definition. pBiself(A, B) is program A expelling from program B as element and B expelling from A as element and the first from the second simultaneously: $\begin{matrix} \mathrm{Sprt}_A^B \\ \mathrm{Sprt}_A^B \end{matrix}$

$\mathrm{Sprt}\begin{matrix} \mathrm{Sprt}_B^A \\ \mathrm{Sprt}_A^B \end{matrix}$.

Definition. pNthself(A_1, A_2, …, A_N) is program A_j expelling from program A_i as element, ∀j, i = 1, 2, …, N simultaneously.

Definition. pSTRself(A) is any part Q of program A expelling from any other part of A as element simultaneously.

Definition. pSTpRself(A) is any part Q of program A expelling from any other part of A as element partially and simultaneously.

Definition. pSTp$_1$Rself(A) is any part Q of program A partially expelling from any other part of A as element simultaneously.

Etc.

On base of these Definitions we can use Nth-programs.

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 «ΛΌΓΟΣ» 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 Pasynkov 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: 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
4) Danilishyn I., Danilishyn O. DYNAMIC SETS THEORY: SIT-ELEMENTS AND THEIR APPLICATIONS. Preprint. Research Square. *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

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 WISSENCHAFTLICHEN 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 «ΛΌΓΟΣ» 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 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

15) 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. Introduction to Dynamic Operators: Lprt-Elements and Applications to Physics and Other their Applications. https://www.opastpublishers.com/journal/journal-of-mathematical-techniques-and-computational-mathematics/articles-in-press

16) Illia Danilishyn, Oleksandr Danilishyn. Introduction to Dynamic Mathematics: Zprt-elements and Their Applications. American Journal of Mathematical and Computer Applications, Volume 1 | Issue 1 (2025), pp.1-146. https://cskscientificpress.com/articles_file/278-_article1742642034.pdf

17) Illia Danilishyn, Oleksandr Danilishyn. Mathematical fundamentals of constructing pseudoliving energy theory and dynamic programmings: monograph / ed. by Pasynkov V. – Shawnee, USA: Primedia eLaunch LLC. – 2025. – 382 p. ISBN 979-8-89660-275-0 DOI 10.36074/mfocp-letadp.monograph-2025. https://dynamical-math.com/materials.html

18) Illia Danilishyn, Oleksandr Danilishyn. Mathematical uncertainties and their applications: monograph / ed. by Pasynkov V. – Shawnee, USA : Primedia eLaunch LLC. – 2025. – 612 p. ISBN 979-8-89660-284-2. DOI: https://doi.org/10.36074/muata.monograph-2025 https://dynamical-math.com/materials.html

19) Illia Danilishyn and Oleksandr Danilishyn. Elements of Dynamic Science. Journal of Modern Classical Physics & Quantum Neuroscience. Vol:1,3. Pg:1-35. DOI: doi.org/10.63721/25JPQN0112. https://wmjournals.com/physics.html

20) I. Danilishyn, & O. Danilishyn, (2025). Elements of dynamic science: monograph. *Primedia ELaunch LLC*, 573. https://doi.org/10.36074/eods.monograph-2025

21) Galushkin A. Networks: principles of the theory. Hot line-Telecom. M.,2010 (in Russian).

22) Illia Danilishyn, Oleksandr Danilishyn. Introduction to Dynamic Mathematics: Zprt-elements and Their Applications. American Journal of

Mathematical and Computer Applications, Volume 1 | Issue 1 (2025), pp.1-146. https://cskscientificpress.com/articles_file/278-_article1742642034.pdf

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 «ΛΌΓΟΣ» 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. *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

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 WISSENCHAFTLICHEN 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 «ΛΌΓΟΣ» 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 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

15) 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. Introduction to Dynamic Operators: Lprt-Elements and Applications to Physics and Other their Applications. https://www.opastpublishers.com/journal/journal-of-mathematical-techniques-and-computational-mathematics/articles-in-press

16) Illia Danilishyn, Oleksandr Danilishyn. Introduction to Dynamic Mathematics: Zprt-elements and Their Applications. American Journal of Mathematical and Computer Applications, Volume 1 | Issue 1 (2025), pp.1- 146. https://cskscientificpress.com/articles_file/278-_article1742642034.pdf

17) Illia Danilishyn, Oleksandr Danilishyn. Mathematical fundamentals of constructing pseudoliving energy theory and dynamic programmings: monograph / ed. by Pasynkov V. – Shawnee, USA: Primedia eLaunch LLC. – 2025. – 382 p. ISBN 979-8-89660-275-0 DOI 10.36074/mfocp-letadp.monograph-2025. https://dynamical-math.com/materials.html

18) Illia Danilishyn, Oleksandr Danilishyn. Mathematical uncertainties and their applications: monograph / ed. by Pasynkov V. – Shawnee, USA : Primedia eLaunch LLC. – 2025. – 612 p. ISBN 979-8-89660-284-2. DOI: https://doi.org/10.36074/muata.monograph-2025 https://dynamical-math.com/materials.html

19) Illia Danilishyn and Oleksandr Danilishyn. Elements of Dynamic Science. Journal of Modern Classical Physics & Quantum Neuroscience. Vol:1,3. Pg:1-35. DOI: doi.org/10.63721/25JPQN0112. https://wmjournals.com/physics.html

20) I. Danilishyn, & O. Danilishyn, (2025). Elements of dynamic science: monograph. *Primedia ELaunch LLC*, 573. https://doi.org/10.36074/eods.monograph-2025

21) Illia Danilishyn, Oleksandr Danilishyn. Introduction to Dynamic Mathematics: Zprt-elements and Their Applications. American Journal of Mathematical and Computer Applications, Volume 1 | Issue 1 (2025), pp.1- 146. https://cskscientificpress.com/articles_file/278-_article1742642034.pdf

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.

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 https://dynamical-math.com/materials.html
2) Illia Danilishyn, Oleksandr Danilishyn. Introduction to Dynamic Mathematics: dynamic sets, dynamic operators and their applications: monograph / ed. by Pasynkov 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 https://dynamical-math.com/materials.html
3) Illia Danilishyn, Oleksandr Danilishyn. Mathematical fundamentals of constructing pseudoliving energy theory and dynamic programmings: monograph / ed. by Pasynkov V. – Shawnee, USA: Primedia eLaunch LLC. – 2025. – 382 p. ISBN 979-8-89660-275-0 https://doi.org/10.36074/mfocp-letadp.monograph-2025
https://books.google.com.ua/books/about?id=j1xSEQAAQBAJ&redir_esc=y
https://explore.openaire.eu/search/publication?pid=10.36074%2Fmfocp-letadp.monograph-2025
https://dynamical-math.com/materials.html
4) Illia Danilishyn, Oleksandr Danilishyn. Mathematical uncertainties and their applications: monograph / ed. by Pasynkov V. – Shawnee, USA : Primedia eLaunch LLC. – 2025. – 612 p. ISBN 979-8-89660-284-2. DOI: https://doi.org/10.36074/muata.monograph-2025
https://dynamical-math.com/materials.html
5) I. Danilishyn, & O. Danilishyn, (2025). Elements of dynamic science: monograph. *Primedia ELaunch LLC*, 573. https://doi.org/10.36074/eods.monograph-2025

https://dynamical-math.com/materials.html

6) Danilishyn I.V. Danilishyn O.V. THE USAGE OF SIT-ELEMENTS FOR NETWORKS. IV International Scientific and Practical Conference ”GRUNDLAGEN DER MODERNEN WISSENCHAFTLICHEN 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. 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**

8) Danilishyn I.V. Danilishyn O.V. DYNAMICAL SIT-ELEMENTS . IV International Scientific and Practical Conference ”GRUNDLAGEN DER MODERNEN WISSENCHAFTLICHEN FORSCHUNG ”, 31.03.2023/Zurich, Switzerland. https://archive.logos-science.com/index.php/conference-proceedings/issue/view/9

9) Danilishyn I.V. Danilishyn O.V. tS – ELEMENTS. Collection of scientific papers «ΛΌΓΟΣ» 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**

10) 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**

11) 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

12) 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

13) Oleksandr Danilishyn and Illia Danilishyn. Dynamic Sets S1et and Some of their Applications in Physics. Science Set Journal of Physics, Volume 2 | Issue 3 25 September 2023,1-11, https://mkscienceset.com/articles_file/255-_article1763988286.pdf I.

14) Danilishyn, O. Danilishyn Dynamic Sets Se, Networks Se. Advances in Neurology and Neuroscience, 2023, Volume 6 | Issue 2 | 278-294. https://www..com/open-access-articles/dynamic-sets-se-networks-se.pdf

15) 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/open-access-articles/program-operators-sit-ts-s1e-set1.pdf

16) Danilishyn I., Danilishyn O. DYNAMIC SETS THEORY: SIT-ELEMENTS AND THEIR APPLICATIONS. Preprint. Research Square.МЕТОД СТАТЬЯ

опубликована 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

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

17) 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 «ΛΌГОΣ» 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**

18) 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**

19) 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**.

20) 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 «ΛΌГОΣ» with Proceedings of the IIInternational Scientific and Practical Conference, Seoul,April28,2023. Seoul-Vinnytsia: Case Co., Ltd.& European Scientific Platform, **pp.144-150. https://archive.logos-science.com/index.php/conference-proceedings/issue/view/10**

21) 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)

22) 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**.

23) 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**

24) 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)

25) 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)

26) Danilishyn I.V. Danilishyn O.V. MATHEMATICS ST, PROGRAMMING OPERATORS STAND SOME EMPLOYMENT Комплексний підхід до модернізації науки: методи, моделі та мультидисциплінарність: матеріали IIМіжнародної наукової конференції, м.Луцьк, 3 березня, 2023р. / Міжнародний центр наукових досліджень. — Вінниця: Європейська

наукова платформа, 2023,p.p. 114-119.
https://archive.mcnd.org.ua/index.php/conference-proceeding/issue/view/03.03.2023

27) 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**

28) 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**

29) 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/

30) 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/

31) 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/

32) 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

33) 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)

34) 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

35) 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**

36) Danilishyn I., Danilishyn O. PROGRAM OPERATORS SIT, tS, S1e, Set1. Preprint. Research Square. НАУЧНАЯ СТАТЬЯ *Опубликовано: 2023-08-04 08:00:27* https://www.researchsquare.com/article/rs-3228799/v1 https://doi.org/10.21203/rs.3.rs-3228799/v1

37) 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/open-access-articles/fuzzy-dynamic-fuzzy-sets-variable-fuzzy-hierarchical-dynamic-fuzzy-structures-models-operators-for-dynamic-singular-hier.pdf

38) 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

39) Oleksandr Danilishyn and Illia Danilishyn. Introduction to Section of Dynamic Mathematics: Dynamic Sets Theory: Parallel Sprt-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-sprtelements-and-their-applications.pdf

40) Oleksandr Danilishyn and Illia Danilishyn. Introduction to Section of Dynamic Mathematics: Dynamic Fuzzy Sets Theory: Parallel Fuzzy Sprt-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/open-access-articles/introduction-to-section-of-dynamic-mathematics-dynamic-fuzzy-sets-theory-parallel-fuzzy-sprt--elements-and-their-applica.pdf

41) Oleksandr Danilishyn, Illia Danilishyn and Volodymyr Pasynkov. Intrtoduction to Section of Dynamic Mathematics: SCprt – elements and Their Applications to Physics. Sci. Set. J of Physics Volume 3 | Issue 5 (2024), pp. 1-23 https://mkscienceset.com/articles_file/838-_article1763989352.pdf

42) 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/open-access-articles/introduction-to-dynamic-operators-rprtelements-and-their-applications-rprtnetworks-variable-fuzzy-hierarchical-dynamic-f.pdf

43) 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. Intelligencehttps://www.opastpublishers.com/journal/advances-in-machine-learning-artificial-intelligence/articles-in-press DOI: https://doi.org/10.33140/AMLAI.05.04.02.

44) 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

45) Oleksandr Danilishyn, Illia Danilishyn and Volodymyr Pasynkov. Introduction to Dynamic Operators: SDS-Elements, Self-type SDS -Structures and their Applications to Physics Sci. Set. J of Physics, Volume 3 | Issue 6 (2024), pp.1-15. https://mkscienceset.com/articles_file/473-_article1763989441.pdf

46) 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

47) 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 10 (2024), pp.1- 16. https://www.opastpublishers.com/open-access-articles/introduction-to-section-of-dynamic-mathematics-theory-of-singularities-of-the-type-synthesizing.pdf

48) Illia Danilishyn, Oleksandr Danilishyn. Introduction to Dynamic Mathematics: Zprt-elements and Their Applications. American Journal of Mathematical and Computer Applications, Volume 1 | Issue 1 (2025), pp.1-146. https://cskscientificpress.com/articles_file/278-_article1742642034.pdf

49) Illia Danilishyn, Oleksandr Danilishyn. Introduction to Dynamic Mathematics:
Beginnings of mathematical uncertainties theory. J Math Techniques Comput Math (*Journal DOI: 10.33140/JMTCM*), Volume 4 | Issue 9 (2025)

50) Illia Danilishyn and Oleksandr Danilishyn. Elements of Dynamic Science. Journal of Modern Classical Physics & Quantum Neuroscience. Vol:1,3. Pg:1-35.
DOI: doi.org/10.63721/25JPQN0112. https://wmjournals.com/physics.html
https://dynamical-math.com/materials.html

S C I E N T I F I C E D I T I O N

Illia DANILISHYN
Oleksandr DANILISHYN

MATHEMATICAL FUNDAMENTALS OF INTERPRETATIONS THEORY (TYPES OF FUTURE SCIENCE) AND FUNDAMENTALLY NEW APPROACH TO ELEMENTARY PARTICLES, PHYSICS, BIOLOGY AND DYNAMIC PROGRAMMING BY PSELF-TYPE AND PASELF-TYPE

MONOGRAPH

Edited by Volodymyr Pasynko

Date of publication: 20.01.2026.
The volume of data is 6,5 MB.
Conventionally printed sheets 19,53.

Publisher: Primedia E-launch LLC.
KS 5518, United States, Kansas, Shawnee, Flint St.
E-mail: info@primediaelaunch.com
URL: www.primediaelaunch.com