Structural Emergence in Partially Ordered Sets is the Key to Intelligence

Sergio Pissanetzky

Graduate School. Dep. of Physics. Texas A&M University
College Station, Texas 77843, USA. Retired
sergio@SciControls.com http://www.scicontrols.com

Abstract. Extraordinary structural organization known as *emergence* is observed in partially ordered sets when a recently discovered functional is minimized. Emergence creates the first structures, and *feedback* reuses them to create hierarchies of structures. The partially ordered set is the *knowledge representation*, the functional connects local behavior to global phenomena, emergence and feedback correspond to *inference*, and the structures and hierarchies to *objects* and *inheritance hierarchies*. If intelligence includes the ability to solve problems, then the structures represent intelligence and emergence represents the build up of intelligence. Since the structures are mathematically obtained from first principles, the finding is proposed as an explanation for the origin of intelligence, and the functional as the key for AGI. Three previous computer experiments, and another one reported here, duplicate higher functions of the human brain and confirm the findings.

Keywords: AI, AGI, emergence, complex systems, brain, refactoring, object-oriented, parallel programming

1 Introduction

The phenomenon of *emergence* is frequently observed in many types of complex dynamical systems when *structures* unexpectedly form in the course of evolution of the system. Despite many years of intense research, the phenomenon of emergence remains unexplained, and a causal relationship between the properties or interactions of the components of the system and the resulting structures has not been found. There is no comprehensive theory of emergence or suitably fundamental model within which to situate emergence [1]. A phenomenological characterization of emergence including precise terminology is available [2].

The original motivation for the present work was the author's interest in *refactoring* [3]. Refactoring is practiced by every software developer virtually all of the time. The term was introduced for object-oriented (OO) code, but it was soon extended to non-OO code [4] and non-software systems such as the law [5], databases [6], and even bacteria [7]. Refactoring is a universal phenomenon, and we all practice it all the time, for tasks ranging from preparing our daily schedule to writing a scientific paper or a theory of Science. Refactoring, however, has

always been considered as something that only humans can do. Tools have been developed for software, of course, but they must operate under the guidance of a human and can be modified or expanded only through human intervention. The scope of refactoring is vast, and it has so far resisted full automation.

Work started with the publication in 2007 of the *imperative* form of the *Matrix Model of Computation* (iMMC) [8], a universal virtual machine that interfaces easily with software and supports refactoring transformations. The iMMC consists of a *matrix of sequences* and a *matrix of services*, both *sparse* [9]. Soon it was noticed that certain *canonical* submatrices of the matrix of services had extraordinary self-organizing properties [10], and the *canonical* form of the MMC (cMMC) and the Scope Constriction Algorithm (SCA) were introduced. SCA uses a *functional*, defined over the set of symmetric permutations of the canonical matrix. The functional assigns a *cost* to each permutation, not necessarily unique, and SCA finds the subset of permutations with the minimum cost. Remarkably, the matrices in the subset are *organized* and contain *structures* that did not exist in the original unpermuted matrix, even though nothing had been done to achieve such result. This is a purely mathematical result. Applications to refactoring [11] and image recognition [12] were published.

A detailed analysis of the inner workings of SCA was then undertaken, and resulted in publication [13], where general transformations from software to the cMMC were proposed, the basics of MMC supervised learning were covered, and an extensive case study on refactoring was included, where a Java program was converted to C (manually) and randomly rearranged to remove all OO features and organization, and automatically refactored by SCA, resulting in objects similar to the original ones. It became clear that a very strong connection existed between the canonical model and AGI.

At that point, it was noticed that a one-one correspondence existed between canonical matrices and partially ordered sets, and that the properties being studied were indeed properties of partially ordered sets, one of the most prevalent and fundamental structures in Mathematics. That explained the vast scope previously observed, and led to publication [14], where the fundamental mathematical principles underlying the observed properties were anticipated, experiments were discussed, and the hypothesis was advanced that those same principles could be used to explain emergence and intelligence.

The present work expands on the theory and claims that: (1) a system susceptible of mathematical analysis can be represented as a partially ordered set, where the nature of the elements is irrelevant; (2) a partially ordered set has a *natural structure*, which depends on and is determined by the set and order alone; (3) the natural structure can be found by the minimization of a universal functional, proposed in this paper; and (4) the structure is, in turn, a partially ordered set with structure of its own, giving rise to *feedback* and resulting in a *hierarchy* of structures. Experimental evidence is discussed, and the parallel computer simulation in Section 3 provides a stunning demonstration of the progressive build up of intelligence by inference and feedback, which can actually be seen in each iteration.

2 Theory

A partial order is a set of precedence relations. Any mathematical expression implies a set and a partial order. For example $a = f(b, c)$ implies set $\{a, b, c\}$ and establishes b and c as *predecessors* of a, meaning they must be given before a can be calculated. The notation is $b < a$, $c < a$, where "$<$" is read "precedes."

Procedures for describing a system as a partially ordered set are now examined. Consider first any system amenable to mathematical analysis. The first step in the analysis is to create a mathematical model, introduce variables to describe the model, and write equations describing constraints, interactions and laws of evolution. But the variables and equations define not only the model but also a partially ordered set. In fact, the variables form the set, and the equations establish a partial order among them. The order only states the well known fact that some variables are dependent and others independent.

The task of describing a system as a partially ordered set is simplified when a *computer simulation* is available. The simulation code provides a display of the equations and required calculations in full detail, with all the information about variables and precedence. It should be possible to develop a parser to automate the conversion. The author has developed one for single-assignment C.

Sometimes, a mathematical description is not even necessary. For example, the brain can be considered as a set of neurons with a partial order defined by their synaptic connections: neurons A, B and C precede neuron D if the simultaneous firing of A, B and C causes D to fire. The idea is expanded at the end of this Section.

The theory is presented next with the help of a simple example. Let S be a finite set and ω a partial order on S, for example:

$$\begin{aligned} S &= \{a, b, c, d, e, f\} \\ \omega &= \{a < c,\ b < c,\ c < f,\ d < e,\ e < f\} \end{aligned} \tag{1}$$

The pair (S, ω) fully specifies the problem at hand. The nature of the elements of S is irrelevant. The standard notation for the problem defined by Eq. (1) is $6(ac, bc, cf, de, ef)$, where the number on the left is the size of the set, $|S| = 6$.

A set with n elements has $n!$ permutations. Some are compatible with the partial order, some are not. The compatible permutations are said to be *legal*. Let Π be the set of all legal permutations, and let $\pi \in \Pi$ be one of them. If the elements of the set are numbered in the order in which they appear in permutation π, then the *distance* between two elements in π is the difference between their numbers, and the *cost* $L_\omega(\pi)$ of permutation π relative to partial order ω is twice the sum of the distances between the elements of each relation:

$$L_\omega(\pi) = 2 \sum_{\omega} d(\varepsilon, \varepsilon'), \tag{2}$$

where $\varepsilon, \varepsilon'$ are elements in S and $\varepsilon < \varepsilon'$ is a relation in ω. To simplify notation, the subscript ω will be omitted. The reason for the factor 2 is explained in [13], and it has a profound physical meaning. Cost $L(\pi)$ is a *functional*, a map

from the set of permutations to numbers. For the example of Eq. (1), there are $6! = 720$ permutations, only 20 of which are legal, $\pi = (b, d, a, e, c, f)$ is one of them, the distance from b to c in π is 4, and the cost of π is 22. A search in set $\mathit{\Pi}$ indicates the existence of many minima, some of them local, others global, even if S is small. At each minimum, there exists a set of permutations, say $\mathit{\Pi}_m$ at minimum m, which has the following remarkable property:

> **Proposition 1.** Set $\mathit{\Pi}_m$ is either a permutation group of set S or a generator for a permutation group of S. In either case, it induces a *block system* in set S.

The block system is the structure being sought. A block system is a partition of set S into disjoint subsets called *blocks*. The elements in each block are *equivalent*, because they stay together inside the block (but in any order) for all permutations of $\mathit{\Pi}_m$. The blocks are *invariant* under the action of $\mathit{\Pi}_m$, because they are the same for all permutations in $\mathit{\Pi}_m$. In the block system, the elements of S appear organized and associated, thus creating logical *meaning*. The emergence of the blocks amounts to *inference*, because they represent a new conclusion obtained from the facts expressed by Eq. (1), where the organization and associations did not exist. And all of this is natural and mathematical.

Feedback also arises naturally as a mathematical phenomenon. A block system resulting from Proposition 1 is, in turn, a set (of subsets of S), and has a partial order induced by the original partial order ω. Proposition 1 applies to the block system just as effectively as it did to the original set S, and repeated application of Proposition 1 results in a hierarchy of block systems.

Functional L is *locally defined*. Its value is a *global* property of the system, but the definition is in terms of *distances*, which are local values. L can be minimized by any local process completely unrelated to and unaware of any global effects that the minimization can cause. Any set consisting of elements capable of minimizing some measure of their interactions with their neighbors will also minimize L and produce the global, unintended effect of the emergence of structures. This is precisely where the transition from a local behavior to a global phenomenon takes place.

In view of all of which, it is hereby proposed that the process that finds the block systems is the core process of intelligence, that intelligence finds its origin in that process, and that the core process can be easily implemented on any regular computer. Intelligent systems are *self-integrated* and *indivisible* [14], but a simple aggregate of intelligent systems (a *society*) is not intelligent, because the systems can not integrate. For finite sets, all calculations in this theory are *computable* and *deterministic*. However, they are *unpredictable* in practice for all but the smallest sets. The size of set $\mathit{\Pi}$ is of the order of $n!$, where n is the size of the set. But $79! \approx 10^{80}$, and 10^{80} is the total number of atoms in the universe. Real-world sets are much larger than 79, and predictable calculations are not possible. The present work focuses on small systems, because they must be understood before dealing with larger systems, and they are easy to study without running into computational difficulties.

For the example of Eq. (1), there is only one global minimum with $L = 16$, and the following 2 permutations of S are found there:

$$\begin{array}{l}(a\ b\ c\ d\ e\ f)\\(b\ a\ c\ d\ e\ f)\end{array} \tag{3}$$

The block system induced by the 2 permutations in set S is $(a, b)(c)(d)(e)(f)$, which contains only one non-trivial block. The order in which the blocks appear, and the association between a and b in the first block, did not exist in Eq. (1). They represent the build up of intelligence in the first iteration.

To every permutation of a partially ordered set there corresponds a *canonical matrix*. The canonical matrix for the system of Eq. (1) under permutation $\pi' = (d\,b\,a\,e\,c\,f)$, which is legal and has a cost of 22, is as follows:

	d	b	a	e	c	f
d	C					
b		C				
a			C			
e	A			C		
c		A	A		C	
f				A	A	C

(4)

The matrix is square, lower-triangular, and sparse [9]. Rows and columns correspond to the elements of S, and appear in the order of permutation π'. Following previous conventions, all diagonal elements contain C. The off-diagonal elements correspond to the partial order ω: if $\varepsilon < \varepsilon'$ is a relation in ω, then element $(\varepsilon', \varepsilon)$ is marked with an A in the matrix. One important property of the matrix is that symmetric permutations that leave all A's in the lower triangle always result in legal permutations of set S.

In the canonical matrix, a line from a C on the diagonal, to an A in the same column, to the C in the same row as the last A, is called a *flux line*. For example, the line from the C in position (b, b), to the A in position (c, b), to the C in position (c, c), is a flux line. The *length* of this flux line measured in cells is 6, which is precisely the cost of relation $b < c$ in permutation π'. It follows that the total length of all flux lines is, precisely, the cost of permutation π', and that the effect of the minimization of the cost is to symmetrically permute the canonical matrix in such a way that the A's are brought as close to the diagonal as possible.

A simple, but viable model of the brain can be developed based on this analogy. If the elements of set S are neurons, and their connections correspond to the relations in the partial order, then the connections correspond to the flux lines and the length of the connections corresponds to the length of the flux lines. But the neurons are known to try to shorten their connections or even migrate in order to preserve resources. When they do that, they also *inadvertently and without purpose* minimize the functional of Eq. (2) and *physically cluster* to form the structures described in this paper. This mechanism can explain both memory and intelligence. The clusters of neurons are called *neural cliques* and their existence has been confirmed [15].

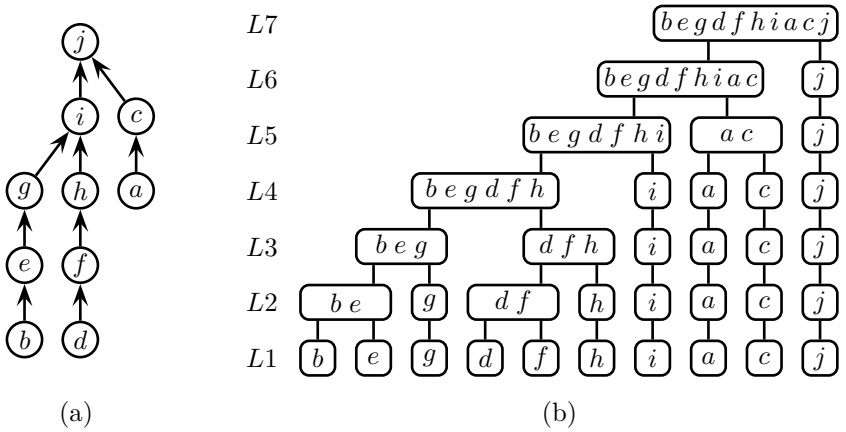

Fig. 1. (a) The solution to the problem of parallel programming of Section 3 and (b) UML diagram of the 7-level hierarchy of objects representing the complete solution for that problem. All structures shown are contained in Eq. (5).

3 Small systems

The experiment for this paper is a simple model of a parallel computer with an undetermined number of CPU's, which has to execute a set of 10 tasks with 9 inter-dependencies. The problem is how to assign the tasks to the CPU's to improve performance. The analysis of 6 different small systems is needed to solve the problem. The first system is Σ_1, specified as follows:

$$\begin{aligned} &\Sigma_1 \ : \ (S_1, \omega_1) \\ &S_1 = \{a, b, c, d, e, f, g, h, i, j\} \\ &\omega_1 = \{a < c, b < e, c < j, d < f, e < g, f < h, g < i, h < i, i < j\} \end{aligned} \tag{5}$$

The 10 tasks are the elements of S and their inter-dependencies are listed as relations in the partial order. This problem is very simple. Any human analyst can solve it in a few minutes. The result is shown in Fig. 1(a). However, the point of the experiment is that the computer can solve the problem *without having been told how to do so*, using only the minimization of the functional. If it does, and if the result is correct, then the claim can be made that the algorithm operates from first principles and is intelligent.

System Σ_1 has 720 legal permutations and a cost range from 28 (with 2 permutations) to 46 (with 180 permutations). It has 2 global minima with 2 permutations having a cost of 28:

$$\begin{aligned} &(b\ e\ g\ d\ f\ h\ i\ a\ c\ j) \\ &(d\ f\ h\ b\ e\ g\ i\ a\ c\ j) \end{aligned} \tag{6}$$

The block system they induce in set S_1 is $\beta_1 = (b\ e)(g)(d\ f)(h)(i)(a)(c)(j)$. This result is illustrated as a UML diagram in Fig. 1(b), where level $L1$ corresponds

to set S_1, and level $L2$ to the block system β_1. The organization of the permutations and the association of b with e and of d with f represent the build-up of intelligence in the first iteration.

As discussed above, the process of structure generation is recurrent. In fact, block system β_1 is itself a set, say S_2, the elements of which are the 8 subsets of which β_1 consists. Set S_2 also has a partial order, say ω_2, induced by the partial order ω_1 of Eq. (5). S_2 and ω_2 define a new system, say Σ_2, as follows:

$$\begin{aligned} &\Sigma_2 \ : \ (S_2, \omega_2) \\ &S_2 = \{a', b', c', d', e', f', g', h'\} \\ &\omega_2 = \{a' < b', b' < e', c' < d', d' < e', e' < h', f' < g', g' < h'\}, \end{aligned} \tag{7}$$

where $a' = (b\ e)$, $b' = (g)$, $c' = (d\ f)$, $d' = (h)$, $e' = (i)$, $f' = (a)$, $g' = (c)$, and $h' = (j)$. System Σ_2 has 8 elements and 7 precedence relations, and corresponds to level $L2$ in Fig. 1(b). It was found to have 126 legal permutations and a cost range of 22 (with 2 permutations) to 34 (with 36 permutations). It has 2 global minima, 4 local minima, 1 global maximum, and no local maxima. The set of 2 permutations at the global minimum is:

$$\begin{matrix} (a'\ b'\ c'\ d'\ e'\ f'\ g'\ h') \\ (c'\ d'\ a'\ b'\ e'\ f'\ g'\ h') \end{matrix} \tag{8}$$

and the block system induced in S_2 is $\beta_2 = (a'\ b')(c'\ d')(e')(f')(g')(h')$. Block system β_2 corresponds to level $L3$ in Fig. 1(b). But block system β_2 is a set with 6 elements, say S_3, and the partial order ω_2 induces into it another partial order, say ω_3. The entire process can be repeated several more times, resulting in the 7-level structure depicted in the figure. As the reader can see, the diagram has a remarkable similarity with the *inheritance hierarchies* used in OO programming. It is proposed in this work that the diagram is, in fact, the rigorous mathematical equivalent of an inheritance hierarchy in OOP. In all, the following 6 small systems are visited by the feedback loop:

System	Levels	System definition	Associations
Σ_1	$L1/L2$	$10(ac, be, cj, df, eg, fh, gi, hi, ij)$	be, df
Σ_2	$L2/L3$	$8(ab, be, cd, de, eh, fg, gh)$	beg, dfh
Σ_3	$L3/L4$	$6(ac, bc, cf, de, ef)$	$beg\|dfh$
Σ_4	$L4/L5$	$5(ab, be, cd, de)$	$(beg\|dfh)i, ac$
Σ_5	$L5/L6$	$3(ac, bc)$	$(beg\|dfh)i\|ac$
Σ_6	$L6/L7$	$2(ab)$	$((beg\|dfh)i\|ac)j$

(9)

In the last column, associations of immediate precedence are indicated by writing the symbols together, such as dfh, while the symbol "|" indicates *parallelism*, an association without precedence. System Σ_3 in the table is in fact the same system defined in Eq. (1) and discussed in Section 2. This example provides a dramatic demonstration of the role of emergence as the source of first intelligence from a fundamental mathematical principle and the build up of higher intelligence. The results at each step of the process are depicted in Fig. 1(b). Iteration 1

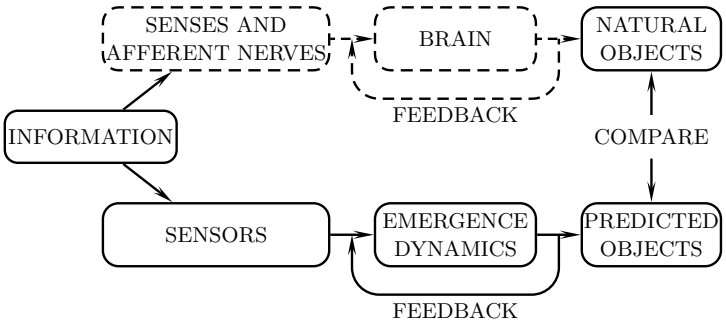

Fig. 2. Computer simulation of higher brain functions. Solid lines indicate easily observable and measurable features, while dashed lines identify features that are not relevant for the present study.

organizes the set and associates be and df, each with immediate precedence and each encapsulated in a separate block. Iteration 2 associates blocks beg and dfh, and iteration 3 associates $beg|dfh$ together without precedence, thus effectively creating a parallel computer with 2 processors. Successive iterations continue reusing previously acquired intelligence to build associations, and the combined effects of inference, feedback and encapsulation are clearly visible. The final solution is identical to the human analyst's solution of Fig 1(a).

4 Experimental evidence

Computer experiments that simulate higher brain functions are easy to perform. The brain itself is irrelevant, and is treated as a black box where only the input and output matter. There exist plenty of carefully documented actual input-output observations with which the simulated results can be readily compared. Figure 2 illustrates the concept.

Information that a person can acquire from the environment, such as a visual image, an observation made by a scientist, or a problem statement received by a computer analyst, is made available as training material for the simulated emergence dynamics (see §IV, "Basics of MMC supervised learning", in [13]). The natural objects that the brain creates are the images we recognize, the theories of Science, or the object-oriented designs the analyst develops. They can be directly compared with the corresponding objects predicted by the simulation.

Three such experiments have been previously published, and one more is reported here in Section 3. The three experiments were further discussed in [14]. The experiments are simple, but they are sufficient for a proof of principle, and they are important because they set directions for future research. The first experiment [11] is in Newtonian Mechanics, and consists of information describing one time step of the motion of a mass particle in three-dimensional space under the action of gravity. A human scientist immediately discovers the following 4 facts: there are 3 components of motion, 2 separate variables must

be used to describe the motion, the 3 components are independent, and the 2 variables are independent. For the simulation, the problem is described as a partially ordered set with 18 elements and 12 relations, and minimization of the functional of emergence immediately discovers a block hierarchy separated into 3 independent components, each in turn separated into 2 independent components, in full agreement with the human scientist.

The second experiment is a case study on refactoring [13], based on a Java program used in many European universities to teach the subject. The program was first converted to C in order to eliminate all object-oriented information, and then represented by a set with 33 elements and a partial order with 55 relations. After minimization of the functional of emergence, a hierarchy of block systems was obtained, which was in excellent agreement with the original Java classes that a human analyst had designed. The conversion of the block systems to Java or C# was not attempted, but it should be easy to automate.

The last experiment is one of image recognition [12], where a set of 167 points distributed on the plane is given. Human observers asked to interpret the image immediately agree that it shows 3 clusters, but disagree when asked to describe the clusters. The number 3 is not explicit in the picture, it has to be found by interpretation. A regular network of cells representing a retina was superposed on the picture. The system was simulated with a set of 1433 elements, and the relations in the partial order were obtained by associating the points with the cells that contained them. Minimization of the functional of emergence yielded a hierarchy of block systems with 3 blocks in the highest level, indicating the separation of the set into 3 clusters, in full agreement with the human observer. In addition, more detailed but not meaningful structure was found in each of the clusters, again in full agreement with the human observer.

One more experiment, the experiment on parallel programming discussed in Section 3 of this paper, is also in full agreement with the human programmer. All four problems have been solved by minimization of the same functional, and by the same algorithm, which is local and knows nothing about the particular problems other than the input S and ω. The ability to solve problems of any kind, when and where they arise, directly from input and without the need for any problem-specific means, is key for intelligence.

5 Concluding remarks

This work has proposed a mathematical theory where a partially ordered set representing a physical system subject to the action of a dissipative dynamical process, naturally gives rise to the phenomena of emergence, feedback, and inference, and becomes self-organized into a hierarchy of block systems where successive levels represent a progressive build up of intelligence. The set serves as a knowledge base with natural support for all the phenomena. The dynamics only dissipates the functional, in a local manner, until exhaustion, and is unaware of the existence of a population or of any global effects that may follow. The functional is universal, defined in terms of the local conditions in the sys-

tem, amenable to be minimized by local dynamics, and providing the only logical connection between the local dynamics and the resulting global phenomena.

Because the theory explains the origin of intelligence from first principles and its growth by feedback and inference, because the features just described are normally associated with intelligence, and because of the supporting experimental evidence, also proposed is the working hypothesis that the theory does describe the origin of intelligence, provides the foundation for a variational theory of intelligence in natural and artificial systems, including the human brain, and allows intelligent behavior to be mathematically described by the elegant principle of optimization.

Important consequences will follow. The value of a variational principle is its unifying power. This work offers many possibilities for new research in AI and AGI, as well as an unprecedented opportunity to unify these fragmented fields.

References

1. Cooper, S.B. Emergence as a computability-theoretic phenomenon. Applied Math. and Computation 215, 1351-1360 (2009).
2. Prokopenko, M., Boschetti, F., Ryan, A. J. An Information-Theoretic Primer on Complexity, Self-Organization, and Emergence. Complexity 15, 11-28 (2009)
3. Opdyke, W. F. Refactoring object-oriented frameworks. Ph.D. thesis, Dep. Comp. Science, Univ. of Illinois, Urbana-Champaign (1992).
4. Garrido, A.,Johnson, R. Challenges of refactoring C programs. Proc. International Workshop on Principles of Software Evolution, Orlando, Florida. 6-14 (2002).
5. Wilson,G. Refactoring the law: reformulating legal ontologies. Juris Dr. Writing Requirement, School of Law, Univ. of San Francisco (2006).
6. Ambler, S. J., Sadalage, P. J. Refactoring Databases: Evolutionary Database Design. Addison-Wesley (2011).
7. Chan, L. Y., Kosuri,S., Endy, D. Refactoring bacteriophage T7. Molecular Systems Biology, article number: 2005.0018. Published online: 13 September 2005.
8. Pissanetzky, S: A Relational Virtual Machine for Program Evolution. Proc. 2007 Int. Conf. on Software Engng. Research and Practice I, 144-150 (2007).
9. Pissanetzky, S. Sparse Matrix Technology. Academic Press (1984).
10. Pissanetzky, S. The matrix model of computation. Proc. 12th. WMSCI Conf. IV, 184-189 (2008).
11. Pissanetzky, S: Applications of the Matrix Model of Computation. Proc. 12th. WMSCI Conference IV, 190-195 (2008).
12. Pissanetzky, S: A new Type of Structured Artificial Neural Networks based on the Matrix Model of Computation. In: Arabnia, H. R., Mun, Y., (eds.) The 2008 International Conference on Artificial Intelligence I, 251-257 (2008)
13. Pissanetzky, S: A new Universal Model of Computation and its Contribution to Learning, Intelligence, Parallelism, Ontologies, Refactoring, and the Sharing of Resources. Int. J. of Information and Mathematical Sciences 5, 143-173 (2009)
14. Pissanetzky, S: Coupled Dynamics in Host-Guest Complex Systems Duplicates Emergent Behavior in the Brain. World Academy of Science, Engineering and Technology, 68, 1-9 (2010)
15. Lina, L., Osana, R., Tsiena, J. Z. Organizing principles of real-time memory encoding: neural clique assemblies and universal neural codes. Trends in Neurosciences, 29, 48-57 (2006).