International Journal of Modern Engineering Research (IJMER)
www.ijmer.com Vol. 2, Issue. 5, Sep.-Oct. 2012 pp-3190-3200 ISSN: 2249-6645

# Compositional Basis of Biological Design

Andrew Kuznetsov
*University of Freiburg, ATG:Biosynthetics, Germany*

***Abstract:*** *Asymmetric sophisticated interactions between parts of a system that is embedded in the environment which changes from time to time leads to modularisation and to growing complexity because the environment has also evolved as a result of the activity of its inhabitants. The author has analysed research on modularity in biology and his own results in this field. Examples of a complex system and programmable self-assembling are given. Recognition and exploitation of modularity in artificial and natural systems are demonstrated. A particular role of horizontal gene transfer as a novel hypothesis on biological modularity is given special attention to. A formal model of compositional evolution under supervision of the environment that changes over time is considered. Some examples of the bottom-up design for artificial genetic networks of increasing complexity are obtainable. Different kinds of behaviour for virtual creatures – sperm cells and ova – are shown. Presented case studies may be useful for the development of a common theory of biological design.*

***Keywords:*** *non-linear dynamics, biological complexity, modules, gene transfer, emergences*

## I. Introduction

Modularity is an old concept in biological science, which is based on the ancient belief that modularity was a real and universal property of Nature. In the 18th century, comparative anatomists such as Georges Cuvier and Geoffroy Saint-Hilaire identified structural modules representing parts of organisms. Joseph Needham in the 1930’s proposed that development can be decomposed into separate elements [1]. In modern times, Walter Fontana, Günter Wagner, Uri Alon, and many others have contributed much to the field [2-6]. Here is a list of recently discovered rules:

- A constant environment that does not change over time leads to non-modular structures
- In contrast, the modular structure can spontaneously emerge if environment changes over time
- Variability in the natural habitat of an organism promotes modularity
- Modularity can also dramatically speed up evolution
- Adaptation of bacteria to new or changing environments is often associated with the uptake of foreign genes through horizontal gene transfer (HGT).

My own research suggests that HGT is the important force, which contributes significantly to modularity [7]. The living systems contain detectable modules such as gene clusters, protein domains, organelles, cells, and tissues and organs with specific biological functions. Just as Richard Watson has written: “the existence of modularity in Nature is now becoming testable” [8], the modularity was revealed in complex biological networks and regulation systems [9-13]. In particular, correlation in high-throughput data and phylogenetic profiles were used for detecting functional modules [14-16].

The main goals of this paper are (1) to assign a role of modularization as the minimalist economic principle for the adaptation to variable environment, (2) to stress the significance of the interaction between modules, and (3) to underline the role of horizontal gene transfer (HGT) in adaptation and evolution because HGT was a major effort in my “evolution by communication” study [17]. We conjecture from previous results that modularity will help to answer the following questions: How can evolution lead to modular systems? Why does modularity exist in biology? Why do complex systems fail or die?

The article is divided into five major parts. The first part is a brief introduction to the field of complexity; the fundamental problem of science with an example of complex system behavior, as well as demonstrations of self-assembly. The second part of the manuscript is about recognition of modules in artificial reality and natural systems. In addition, a case study will be presented of how an inspiration from natural modularity can help resolve real technical challenges. The third part is dedicated to answering a crucial question concerning the origin of modularity based on the analysis of distributed bacterial sulfur metabolism. This section is about the nature of mutations and describes a generalization of copy, cut, and paste mechanisms. The fourth part of the paper looks at the bright possibilities of modular design. It will bring the modularity of genetic networks in *pi*-calculus with examples of elements of networks and genetic motifs. Finally, the ideas of scalable design, emergent behavior, and compositional evolution of biological systems will be considered. Readers will be invited to a speculation about the role of horizontal gene transfer in the biological complexity. The main point presented in this document is the significant role of the transfer and interaction of modules in adaptation, which is still in the shadow of the general evolution paradigm. An example of the program code is included in the Appendix.

## II. Complexity

Let me begin with a bit of philosophy. A fundamental problem in science is why matter grows in complexity. As Herbert Simon wrote in the introduction to the book ‘*Modularity: understanding the development and evolution of natural complex systems*’ [18], “Complexity arises then … components interact with each other in ways ... more than uniform, frequent elastic collisions. Interactions among components can lead to all kinds of nonlinear behavior.” The phase space of complex systems usually exhibits irregular surfaces of local minima and maxima, or even demonstrates bifurcations and chaotic behavior. According to Simon, if we take the phrase “survival of the fittest” literally, then the theory of evolution has validity only in a world where maxima are attainable and the paths toward them are discoverable, which is seldom the case in the real world.

As an example, I wish to present some amazing data obtained when I was a student, the divergence of astrocytes for GFA content depending on malignation [19, 20]. GFA, which means Glial Fibrillary Acidic Protein, is a neurospecific protein, which is a marker of nervous cells such as astrocytes. It is the biomarker of the neoplasia of the Central Nervous System (CNS). We investigated a distribution of GFA protein for different cancer cells in CNS. Because there was a divergence of nervous cells for GFA protein content depending on malignation, which was not easy to diagnose, the linear and nonlinear regression functions were applied gradually. Distributions of soluble and insoluble forms of GFA protein in the normal and malignant nervous cells were finally well described by the canonical cusp catastrophes, i.e. Whitney cusp surfaces (Figure 1).

Unfortunately, ordinary mathematical methods no longer lead to solutions in closed form and moreover, the complexity can carry us beyond the simulation capacities. Examples of 3D and 2D fractal structures in Figure 2 are results of the diffusion limited aggregations (DLA) generated within a computer. Figure 3 presents screenshots of the assembling by adhesion rules. The exercise was to form a chessboard pattern in the presence of DLA. The seed of crystallization is in the bottom-left corner. Figure 3a demonstrates a random initial position of tiles and Figures 3b-d present the final solutions for different parameters of diffusion [21]. One can see that it is impossible to create the chessboard pattern without gaps, but Nature does it better. The great problem is to find those tricky rules of interactions between soft living particles, for instance repair rules, examples of robustness, and so on.

With respect to Herbert Simon's point of view on complexity, I would like to recall his definition of modularity in the terms of interactions. He wrote: "… the frequencies of interaction among elements in any particular subsystem of a system are an order of magnitude or two greater than the frequencies of interaction between the subsystems. We call this … nearly decomposable (ND) system." [22]. In other words, ND systems are made up of separate parts where there is far more relations within each part than between different parts. ND is not the same as modularity but it gives a clue about an essential property of any modular system; this model is very general. Simon continued later: "A system may be characterized as modular to the extent that each of its components operates primarily according to its own, intrinsically determined principles. Modules within a system or process are tightly integrated but relatively independent."

## III. Recognition of modules

Examples of modularity are quite natural in today's software design, everyday engineering practice like electronics, optics, and even in DNA-nanotechnology such as DNA-origami building blocks, modular DNA folding, and DNA-protein interactions in the case of binding complex GCN4 bZIP with DNA [23]. A modular design facilitates development and maintenance of complex technical devices. In natural science, modularity is represented by a search for fundamental units and basic elements, for instance, elementary particles, chemical elements, molecules and compounds, gene clusters, metabolic pathways, etc. The search for modularity is the identification of sets of base elements and construction rules to recognize simplicity in complex systems.

Figure 4 demonstrates an example in nanotechnology, the result of computer added design of endohedral metallofullerenes on the basis of quantum mechanical calculations within the density function theory (DFT). Cobalt-clusters and carbon-fullerenes were considered as independent modules which can be in different positions and could interact in different ways. We were able to use this approach to plan experiments rationally and to find an empirical rule for the interaction between Co- and C-atoms that describes the magnetic moment of the complex as a function of number and length of chemical bonds: the magnetic moment $M$ per Me-atom of given complex is proportional to the average Me–C bond length $L$ divided by the total number $N$ of Me–C bonds in the complex [24, 25].

***2.1. What is a module?*** As was shown above, the idea of modularity is intuitive but is hard to formulate. To give an example; Uwe Strähle and Patrick Blader wrote: "... we define a module as an assembly of biological structures that fulfill a function in an integrated and context insensitive manner. Function as defined here is not merely the interaction of molecules but an interaction that yields a biological output which is characteristic of the module. Furthermore, the application of the module is flexible. To be recognized as a module, it has to be used either in different processes in the same organism or in different organisms, exploiting its invariant functional properties in the same or different processes. A module is therefore characterized by its reiterated use." [26]. It looks rather difficult. I am sure some readers have recognized this problem. In view of Uri Alon having defined modularity in a more laconic way; a "property of a system which can be separated into nearly independent sub-systems" [27]. In other words, modularity is defined through a process that starts by recognizing patterns, shapes, or events that are repeated at some scale of observation. Modularity is a hallmark of biological organization and an important source of evolutionary novelty. Modularity is a sign of the universal principle of economy in Nature.

Biological systems present both genotypic and phenotypic modularity. Most biological functions are carried out by particular groups of genes and proteins so that one can split the structure into functional modules. For example, proteins work in groups, such as complexes and pathways. I like this simple definition: *a module is a set of genes that act together to carry out a specific function*. The modularity of biological networks is puzzling and the recognition of modularity came as a surprise. In this situation: (1) find modules, relations between modules, the origin of modules, (2) understand the hierarchy of a modular system and a reason of the entanglement within modules and between modules because modularity is the basis of our ability to separate problems into smaller parts that can be studied independently to assign functions to genes, proteins, metabolic and signaling pathways. In my opinion, the answers to the following questions could have given a key to control an evolutionary process: (1) How does a system evolve and fail? (2) What is the limit of evolvability? In addition, I would like to point out that *evolvability* is the ability to respond to a challenge by producing the correct variation.

***2.2. Natural modularity (dsr and sox gene clusters).*** I investigated *dsr* and *sox* gene clusters coding enzymes for bacterial community sulfur metabolism (*dsr* means the dissimilatory sulfate reduction and *sox* means the sulfide oxidation). Studies on environmental DNA databases such as Community Cyberinfrastructure for Advanced Marine Microbial Ecology Research and Analysis – CAMERA for short, allowed me to confirm the major role of HGT in modularity. The correlation between the *dsr* and *sox* clusters for the experimental set of 41 stations around the world was R = 0.86, which demonstrates the complementarity of *dsr* and *sox* metabolic pathways in environmental populations. Genes tend to group in modules that facilitate the propagation of specific functions within a community of organisms. To confirm this observation, the AprA tree was produced using *A.pompejana* symbiont reference and NCBI BLAST pairwise alignments. However, it was impossible to map AprA tree to 16S rRNA phylogenetic tree because putative HGT affected the canonical phylogenetic tree. I revealed a hierarchical modularity in the bacterial sulfur metabolism. The example included a repressor of phase-1 flagellin within the large sulfite reductase 4Fe-4S domain [7].

## IV. Origin of modularity

According to the Günter Wächtershäuser, William Martin, and Eugene Koonin scenario of the origin of life [28, 29], the sulfur metabolism was a very ancient invention of Nature. The data obtained from modern DNA-sequence databases allowed me have a look back 3.5-3.8 billion years, when unicellular life emerged on our planet. This logic helped me to merge the origin of life with the origin of modules. Modularity could be explained as an adjustment for evolvability. I concluded that modularity is an unavoidable design feature of organic life [7]. Evidences of HGT and modular nature of prokaryotic genomes are very good news for engineers who are trying to design complex metabolic pathways. Such artificial modular designs might include a kernel (housekeeping genes) and modules involved in a process of adaptation toward a particular environment. Travelling groups of genes could be easily embedded in almost any system.

***3.1. Nature of mutations.*** Unfortunately, biological reality is much too complex to be captured by a linear mapping of genes to phenotypes. However, it is a reasonable mechanistic assumption that the genome produces adaptive variants. Such a genome is able to get access to regions of the phenotype that may be adaptive in two ways: (1) by the mechanisms of compositional evolution which combine interdependent genetic modules that have evolved previously in parallel, and (2) using small gradual changes like point mutations. In other words, it means (1) the change of topology of genetic networks, and (2) the change of parameters which are induced in DNA sequences.

An example from experiments on the CAMERA database is in Figure 5, the alignment of the protein reads from microorganisms living in the gutless worm *O.algarvensis* to the AprA polypeptide from bacterium *V.okutanii*. I was able to show different patterns of variability for AprA protein [7]. Numerous exchanges, deletions and insertions in AprA polypeptide were found as are marked in yellow.

***3.2. Modularity as a set of construction rules, the cut and paste Argo-machine.*** An extreme specialization is somewhat risky for an organism in diverse conditions. A remodularization of the organism is needed in the changing environment. Consider an evolving system – an abstract machine and an environment that is continuously changing creates advice words for the machine to stimulate an adaptation of this device to its surroundings (Figure 6a). The input for this machine is special words which are generated by the environment. We call these input words 'oracle', 'guide', or 'generative'. The output is phenotypes which fit the environment. The machine operates on strings, which can code organisms in hierarchical manner. This non-deterministic abstract machine searches according to oracle words in the design space proper to its environment by cutting, transposing, and pasting a set of tapes [30].

The schematic of *Argo-machine* (***AM***) with circular tapes is shown in Figure 6d. The ***AM*** consists of agents and each of these has a head and a tape and can be in different output states. In general, the tape is a nonempty string of symbols that may be linear or circular. The head scans the tape according to an input word $w_i$ and cuts it at recognized sites. The agent arbitrarily pastes the tape. For each tape-configuration there is an appropriate output state of the agent that is checked by the environment. Special 'accept' and 'reject' states take immediate effect. An agent accepts if its output state corresponds to the environment state; an agent will reject if less than two matches to the input word exist on the tape. ***AM*** can accept if at least one agent accepts, reject if all agents reject or loop (Figure 6b). If the environment has changed, it delivers a transposition and a new word $w_{i+1}$. The transposition means to make a copy of the tape from the accepted agent to other ones and join it from head-to-tail (Figure 6d). ***AM*** continually looks for an agreement with the environment.

We could describe ***AM*** in another way. The system operates on inputs and memory, uploads the memory, and yields outputs. Changes in environment generate 'oracle' words which guide DNA shuffling and transpositions. A combinatorial power of this system is very high. In short, ***AM*** is a set of stochastic cut-paste agents, which act in parallel on their own tapes according to the instructions (input words), communicate with each other by transpositions of the tape, and interact with the environment to compare the output states. Based on the comparison it accepts or runs in a loop to fit to the environment.

A crucial point of ***AM*** is the *Argonaut algorithm*. Each agent, Argonaut in other words, performs the next actions on word *w*:
(1) Scans the tape to be sure that it has at least two matches. If not, rejects it.
(2) Cuts at the matching sites and arbitrarily paste the tape's fragments.
(3) Takes the output state according to the new tape.
(4) Checks the output state with the state of the environment. If satisfied, accepts; otherwise loops.

The combinatorial power of the *Argonaut algorithm* is bigger than polynomial, exponential, and factorial functions (Figure 6c). Computation associated with *Argo-machine* is the shuffling of tapes from the initial set $T_0$ until an accepted state is reached – an adaptation. Usually, computation never ends, because the environment changes permanently; if this happens, the case, called a catastrophe, leads to a transposition, generates a super-transition from the accept-state to the set of new initial-states, and brings a new generative word. *A progression of adaptations and catastrophes is an evolution*.

## V. Modularity of genetic networks in *pi*-calculus, a modular "table of elements"

The model in our case is an abstraction that summarizes the property of a modular structure. Some people believe that Nature uses a restricted number of models to establish a set of relations among modules. I used this *à la* Wolfram approach, introduced in 2002 by Stephen Wolfram in his book [31]. In contrast to Wolfram's 1D-cellular automata, my research was based on *pi*-calculus to build genetic networks. The *pi*-calculus was invented in 1992 by Robin Milner with the aim of describing interactive systems [32]. As a next step, the stochastic *pi*-calculus was proposed by Corrado Priami in 1995 [33], while Andrew Phillips from Microsoft developed SPiM language and designed a stochastic simulator (Stochastic Pi-Machine, SPiM for short) that performed a Gillespie algorithm [34, 35]. I used SPiM language to write a code for the SPiM simulator (see Appendix).

The following primitives in SPiM calculus were used to make networks with different topologies: decay (degradation of a transcription factor), *null* gate (constitutive transcription), gene product (protein transcription factor), *neg* gate (negative regulation), *pos* gate (positive regulation). I built genetic circuits of increasing complexity from those five basic primitives. Furthermore, I investigated the circuits such as negative and positive genetic regulators, genetic switches, oscillators, impulse generators, feedback and feedforward loops, genetic memory elements, and bifan motifs.

My work in the 'silicon laboratory' represented the bottom-up scenario of compositional design in terms of SPiM calculus. Most models chosen arbitrarily worked well at standard values of parameters. The topology and complexity of networks played a significant role in their behavior. New networks emerged easily, sometimes because of duplications and transpositions of the earlier ones. These operations resembled natural genetic mechanisms such as vertical and horizontal gene transfer – essential operators of biological evolution. Some variations in parameters were similar to 'point mutations' leading to an optimization (adaptation) of the system to a desirable pattern of behavior [36].

***4.1. Basic genetic gates.*** Figure 7 presents the basic genetic gates, such as negative and positive regulators without input, with regulated input, also the cases of autoregulation. For each plot, an abscissa indicates the time of simulation with an ordinate that is the number of protein molecules. Simulations were started in the absence of proteins by doing a constitutive transcription. The number of protein molecules initially increased along with the time of simulation and finally levelled off at the equilibrium between production and degradation. The constitutive expression and the output were higher for the negative regulator than for the positive one.

To see a response of genetic elements, an input allowed linear increases from 0 to 100 individual molecules then decreased linearly to 0. The input molecules were injected into the system at a certain times. As a result of the reaction, the negative gate behaved like an inverter whereas the positive gate increased the output signal almost 10 times. The results of negative and positive autoregulations are also shown in Figure 7.

***4.2. Repressilator.*** A repressilator consists of three *neg* gates that mutually repress each other as shown on the picture (Figure 8). Simulation of the repressilator at nominal parameters resulted in an irregular duration of protein cycles. The decreased rate of gene unblocking $\eta_p$=0.001 and the increased protein binding $r$ =10.0 allowed an improvement of the regularity of oscillations. Populations of proteins stabilized nearly 100 molecules in each cycle with the same duration of impulses. The program code for this experiment is given in the Appendix.

***4.3. Bi-stability and memory.*** Previous experiences were used to design a genetic memory element. The closed chain from four *neg* elements demonstrated bi-stable characteristics. The system arbitrarily started from expression (*a,c*) or (*b,d*) proteins. Fortunately, the circuit established stable behavior at a standard range of parameters. After it dropped to an arbitrary state, the system survived for a long time (Figure S1a,b, insertions). However, the circuit was sensitive to external inputs. For example, when the system is in state *bd*, then a programmable input *a* can change its state to a new state *ac*, to be exact, the production of *b* and *d* proteins can be changed to *a* and *c* proteins by input *a* (Figure S1c). Moreover, if input *a* no longer exists, the system nevertheless stays at the state *ac* and does not turn back until the specific input *b* changes the system's state again (Figure S1d).

***4.4. Synchronous FFBL.*** I discovered that the coherent feedforward and feedback loops (FFBL) circuit can be in four different states corresponding to particular patterns of protein expression: I, red – low basal production of *b* and *c* proteins, II, green – intensive stable production of *b* and *c*, III, blue – spontaneous synchronous outputs of *a* and *c*, and IV, black – exhaustive expression of *a* and *c* proteins with gaps. These essential states were significantly influenced by stochastic fluctuations. Nevertheless, I did not observe any transition from one state to another when the parameters were fixed (Figure S2).

***4.5. Asynchronous FFBL.*** An incoherent FFBL circuit showed even more sophisticated outputs. Sometimes the system demonstrated unstable dynamic behavior. I found this circuit in the following states: I, red – decoherent small amounts of *b* and large amounts of *c* proteins, II, green – low level of *b* protein and intermediate level of *c* protein, III, blue – flip-flops between *a* and *c* production, and IV, black – asynchronous low *c* and intermediate *a* proteins expression. In addition, I introduced the next states: Ia, orange – for low *b* with intermediate and high *c* levels, as well as state IIa, yellow – for intermediate *b* and high *c* proteins expression. It should be remembered that synchronous and asynchronous FFBL are the common features of real genetic networks (Figure S3).

## VI. Scalable design

***5.1. Compositional mechanisms of modularity; interaction, communication.*** As I mentioned, instead of small gradual changes like point mutations, the mechanisms of compositional evolution combine interdependent genetic modules that have evolved previously in parallel. Examples of compositional mechanisms in Nature include recombination, hybridization, symbiotic encapsulation and horizontal gene transfer (HGT), as exhibited in the history of major evolutionary transitions. Modules must persist as identifiable units to be assembled into a Goldschmidt's 'hopeful monster' [37]. Different species are constantly exchanging genes, often with viruses as the messengers. Microbes can pass on fragments of DNA to each other during horizontal gene transfer. Even an entire pathway can be transferred if the respective genes are placed close to each other on the DNA sequence [38]. Furthermore, 'travelling' pathways exert their functions in cells with different genetic background and in changing environments.

Interspecies gene transfer also occurs (at an unknown rate) among more complex species, including humans. We demonstrated HGT in some eukaryotic species, such as mussels [39], fish [40], and rabbit [41] in lab conditions. Figure 9 shows the result of successful pcDNA3-*lacZ* sperm-mediated gene transfer into fish *M.fossilis*. In that case, a combination of electrical impulses with dimethyl sulfoxide (DMSO) treatment was used to improve the efficacy.

***5.2. Design of complex systems: make parts, repeat them, and change them.*** It is well known that recursive functions generate fractals (Figure 2), less known are recursive functions in agents. Unfortunately, collective behavior and interaction between agents have been mostly ignored by biochemists and molecular biologists. Here, I give an example of a simple system that can produce a complex behavior.

An artificial world of Sperm Cells and Ova was investigated in the agent-based simulation [42]. If the meeting of a Spermatozoon and Ovum leads to a new Spermatozoon and new Ovum with a new genome, then 'genome mutations' will have occurred. This system demonstrates different kinds of behavior depending on the 'mutation' parameter *R*.

In detail, each creature has a circular genome consisting of 1024 'genes', only one of them is active and color coded with mod1024. The state of each creature is described by following recursive function:

```
T(i+1) <- ([T(i) + P(i)] / 2 * R)mod1024
P(i+1) <- T(i+1),
```

where T(*i*) is the color code of the individual Spermatozoon and P(*i*) is the color code of the individual Ovum at the time *i* of breeding. R is the mutation parameter on the interval ]0, 4].

***5.3. Emergent behavior depending on mutation parameter.*** The system demonstrated ordered (R<=1) and complex (R>1) regimes, such as (1) stable focus, R=1, (2) periodic, R=1.01, and (3) chaotic, R=3 regimes, as well as (4) strange attractor, R=4 (Figure 10). This complex and unexpected behavior of the artificial world of two agents – Sperm Cells and Ova – appeared from the collective dynamics of the distributed creatures and parallel execution of the recursion.

## VII. Conclusion

Search for modularity in Nature is similar to pattern recognition, a native ability of the human mind. However, the discovery of hidden rules and algorithms leading to modularity is not an easy mission. The first attempt to describe evolutionary processes in terms of modules was carried out by John Holland in his building block hypothesis [43]. Richard Watson introduced three different algorithmic paradigms of evolution [8]. Watson classified systems on the basis of interdependency of variables. He considered weak, modular and arbitrary interdependencies of variables; smooth, spike and ruffle fitness landscapes; different optimization methods such as hill-climbing, divide-and-conquer decomposition, exhaustive and random search; different kinds of complexity on the basis of the number of variables and the number of values for each variable. Finally, Watson provided an evolutionary analogy for each class that is to say gradual evolution, compositional evolution, and impossible analogy or 'intelligent design'.

The author of this manuscript has seen the origin of modularity in a specific interaction between components of complex systems. *Exchange of modules has appeared as a formal origin of living entities.* It is my firm belief that interaction between a given system or multiple systems (agents) and an environment leads to the change of this environment along with a new adaptive behavior of its inhabitants. These kinds of interactions lead to a progressive increasing complexity of the system and environment as a result of more sophisticated interactions between components, see [44]. In this full of life situation resulting from the 'Red Queen Effect', a natural *de novo* design is preferable in the sense of economy than a possible reconstruction of 'old' creatures. I expect this imaginable picture is a possible response to the great question of why complex systems finally fail or die.

An astonishing observation on the global bacterial sulfur metabolism that the lateral gene transfer affected 16S rRNA phylogeny leads me to the following important conclusion: "the lateral gene transfer supports the modularization on the global scale" with the corollary: "the recombination provides modularization in protein structures". I have explained in this paper the model of the abstract *Argo-machine*, which is driven by 'oracle' words that are generated in turn by the environment. The model was inspired by data on Argonaute proteins and siRNA/RISC complex [45], also by ping-pong amplification loop mechanism and the existence of transposon-rich piRNA genome clusters [46], by rampant horizontal gene transfer in prokaryotes [47, 48], as well as by my own experiments on sperm-mediated gene transfer [39-41], and the latest achievements in the field of SMGT [49, 50]. The view of molecular genetics and epigenetics mechanisms like the 'molecular computation' is in my opinion a very fruitful concept. A good example is research on developmental genome rearrangements in ciliates provided by Landweber and Kari [51]. A few simple rules and algorithms embedded in autonomous agents can lead to a broad variety of the system behaviour that is also demonstrated within this paper in terms of an agent-based approach. The problem of interaction between modules is very important in the practice of genetic engineering because of the desirable compatibility between modules in synthetic design. That was the reason to discuss xenologs versus orthologs in my research on sulfur metabolism in environmental bacterial populations [7].

Before I finish, I would like to summarize the key points:

- A module is the part which operates independently of other components in the system
- Functional modularity is independence in space and time
- Modularity is driven by the interaction and communication of components
- A set of modules can be joined in different ways when the environment changes (e.g. HGT)
- Origin of modularity is in the compositional evolution
- Modularity expands parallel development and enhances evolvability
- Specific interaction between modules is a subject of compositional design of complex systems
- Modularity is the relationship between the whole and the parts.

I believe that the understanding of evolution as a computational process of an ever changing environment can help us find design principles of biological systems. Modular systems consist of subsystems that work autonomously and exert specific functions. Biology can be described in terms of modules; furthermore modularity could be considered as a scientific issue, because modular and hierarchical structures demonstrate evolutionary benefits [52].

## Acknowledgments

The author is solely responsible for any unconventional conclusions presented in this manuscript. In addition, I would like to express my gratitude to Bert Schnell, Heinz Eikmeyer, Mikhail Kats, Genaro Juarez Martinez, Vladik Avetisov, Irina Shchit, Irena Kuznetsova, Sergey Golutvin, Polina Tereshchuk, Svetlana Santer, Konrad Diwold, Martin Schneider, Andrew Phillips, Steven Benner, and many others who are not on the list.

## References

[1] Needham J. On the dissociability of the fundamental processes in ontogenesis // Biological Reviews. 1933; 8: 180-233.
[2] Andrews J. Bacteria as modular organisms // Annual Review of Microbiology. 1998; 52: 105-26.
[3] Ancel LW, Fontana W. Plasticity, evolvability, and modularity in RNA // J Exp Zool. 2000; 288(3): 242-83.
[4] Bolker JA. Modularity in development and why it matters to Evo-Devo // American Zoologist. 2000; 40: 770-6.
[5] Kashtan N, Alon U. Spontaneous evolution of modularity and network motifs // Proc Natl Acad Sci U S A. 2005; 102(39): 13773-8.
[6] Wagner GP, Pavlicev M, Cheverud JM. The road to modularity // Nat Rev Genet. 2007; 8(12): 921-31.
[7] Kuznetsov A. Modularity and distribution of sulfur metabolism genes in bacterial populations: search and design // Journal of Computer Science & Systems Biology. 2010; 3(5): 91-106.
[8] Watson RA. Compositional Evolution: The Impact of Sex, Symbiosis, and Modularity on the Gradualist Framework of Evolution // Vienna Ser Theor Biol. A Bradford Book. 2006.
[9] Guimerà R, Nunes Amaral LA. Functional cartography of complex metabolic networks // Nature. 2005; 433(7028): 895-900.
[10] Hartwell LH, Hopfield JJ, Leibler S, Murray AW. From molecular to modular cell biology // Nature. 1999; 402(6761 Suppl): C47-52.
[11] Ihmels J, Friedlander G, Bergmann S, Sarig O, Ziv Y, Barkai N. Revealing modular organization in the yeast transcriptional network // Nat Genet. 2002; 31(4): 370-7.
[12] Jayaswal V, Lutherborrow M, Ma DD, Yang YH. Identification of microRNA-mRNA modules using microarray data // BMC Genomics. 2011; 12: 138.
[13] Liu B, Liu L, Tsykin A, Goodall GJ, Green JE, Zhu M, Kim CH, Li J. Identifying functional miRNA-mRNA regulatory modules with correspondence latent dirichlet allocation // Bioinformatics. 2011; 26(24): 3105-11.
[14] Pellegrini M, Marcotte EM, Thompson MJ, Eisenberg D, Yeates TO. Assigning protein functions by comparative genome analysis: protein phylogenetic profiles // Proc Natl Acad Sci U S A. 1999; 96(8): 4285-8.
[15] Schuster S, Pfeiffer T, Moldenhauer F, Koch I, Dandekar T. Exploring the pathway structure of metabolism: decomposition into subnetworks and application to *Mycoplasma pneumoniae* // Bioinformatics. 2002; 18(2): 351-61.
[16] Segal E, Shapira M, Regev A, Pe'er D, Botstein D, Koller D, Friedman N. Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data // Nat Genet. 2003; 34(2): 166-76.
[17] Kuznetsov A. Evolution by communication: a revision of sperm-mediated gene transfer // Frontiers in the Convergence of Bioscience and Information Technologies: FBIT 2007, Ed. Daniel Howard et al.; Proceedings of the FBIT 2007 International Conference, Jeju Island, Korea, 11-13 October 2007, P. 322-4.

[18] Callebaut W, Rasskin-Gutman D. (Eds.) Modularity: Understanding the development and evolution of natural complex systems // Vienna Ser Theor Biol. MIT Press. 2005.

[19] Kuznetsov AV. Study of physical, chemical, and immunological properties of the glial fibrillary acidic protein // Graduate Thesis. Dnipropetrovs'k University. 1983.

[20] Kuznetsova IV. Glial fibrillary acidic protein of the human brain // Graduate Thesis. Dnipropetrovs'k University. 1984.

[21] Kuznetsov A. Assembling by adhesion rules at the nanoscale // DECOI 2007: Design of Collective Intelligence, International Summer School on Collective Intelligence and Evolution, Amsterdam, Holland, 20-24 August 2007.

[22] Simon HA, Ando A. Aggregation of Variables in Dynamic Systems // Econometrica. 1961; 29(2): 111-38.

[23] Kuznetsov A. Barbie nanoatelier // IET Synthetic Biology. 2007; 1(1–2): 7–12.

[24] Kuznetsov A. Magnetic properties of endohedral complexes $Co_5@C_n$ depending upon the size and symmetry of fullerenes as well as orientation of cobalt cluster // Computational Materials Science. 2012; 54: 204-7.

[25] Kuznetsov A. From carbides to $Co_5$ and $Co_{13}$ metallofullerenes: first-principles study and design // Am J Biomed Eng. 2012; 2(1): 32-8.

[26] Strähle U, Blader P. The basic helix-loop-helix proteins in vertebrate and invertebrate neurogenesis // in Modularity and Evolution. Eds. Gerhard Schlosser and Günter P. Wagner. University of Chicago Press. 2004.

[27] Alon U. An Introduction to Systems Biology: Design Principles of Biological Circuits // Chapman & Hall/CRC Mathematical & Computational Biology. 2006.

[28] Koonin EV, Martin W. On the origin of genomes and cells within inorganic compartments // Trends Genet. 2005; 21: 647-54.

[29] Wächtershäuser G. Before enzymes and templates: theory of surface metabolism // Microbiol Rev. 1988; 52: 452-84.

[30] Kuznetsov A, Schmitz M, Mueller K. On bio-design of Argo-machine // Explorations in the complexity of possible life: abstracting and synthesizing the principles of living systems, Ed. by Peter Dittrich, Stefan Artmann; Proceeding of the 7th German Workshop on Artificial Life (GWAL-7), Jena, Germany, 26-28 July 2006. P. 125-33.

[31] Wolfram S. A New Kind of Science // Wolfram Media, Inc. 2002.

[32] Milner R. Functions as Processes // Mathematical Structures in Computer Science. 1992; 2(2): 119-41.

[33] Priami C. Stochastic pi-Calculus // Comput. J. 1995; 38(7): 578-89.

[34] Phillips A. The Stochastic Pi-Machine // Available from http://research.microsoft.com/~aphillip/spim/ (2007).

[35] Gillespie D. Exact stochastic simulation of coupled chemical reactions // J Phys. Chem. 1977; 81: 2340-61.

[36] Kuznetsov A. Genetic networks described in stochastic Pi Machine (SPiM) programming language: compositional design // Journal of Computer Science & Systems Biology. 2009; 2(5): 272-82.

[37] Goldschmidt RB. Evolution as viewed by one geneticist // American Scientist. 1952; 40: 84-98.

[38] Wren B. Microbial genome analysis: insights into virulence, host adaptation and evolution // Nat Rev Genet. 2000; 1: 30–9.

[39] Kuznetsov AV, Pirkova AV, Dvorianchikov GA, Panfertsev EA, Gavriushkin AV, Kuznetsova IV, Erokhin BE. Study of the transfer of foreign genes into mussel *Mytilus Galloprovincialis* Lam. eggs by spermatozoa // Ontogenez. 2001; 32(4): 309-18.

[40] Andreeva LE, Sleptsova LA, Grigorenko AP, Gavriushkin AV, Kuznetsov AV. Loach spermatozoa transfer foreign DNA, which expression is discovered in the early development stages // Genetika. 2003; 39(6): 758-61.

[41] Kuznetsov AV, Kuznetsova IV, Schit IYu. DNA interaction with rabbit sperm cells and its transfer into ova in vitro and in vivo // Molecular Reproduction & Development. 2000; 56(2): 292-7.

[42] Kouznetsov AV. Toy SMGT // Alife mutants hackingsession on systems and organisms (AMHSO), Rule 110 Winter Workshop, Bielefeld, Germany, 6-13 March 2004.

[43] Holland JH. Adaptation in Natural and Artificial Systems // Ann Arbor, MI: The University of Michigan Press. 1975.

[44] Szöllosi GJ, Derényi I. Congruent evolution of genetic and environmental robustness in micro-RNA // Mol Biol Evol. 2009; 26(4): 867-74.

[45] Parker JS, Roe SM, Barford D. Structural insights into mRNA recognition from a PIWI domain-siRNA guide complex // Nature. 2005; 434(7033): 663-6.

[46] Aravin AA, Hannon GJ, Brennecke J. The Piwi-piRNA pathway provides an adaptive defense in the transposon arms race // Science. 2007; 318(5851): 761-4.

[47] Boto L. Horizontal gene transfer in evolution: facts and challenges // Proc Biol Sci. 2010; 277(1683): 819-27.

[48] Omelchenko MV, Makarova KS, Wolf YI, Rogozin IB, Koonin EV. Evolution of mosaic operons by horizontal gene transfer and gene displacement in situ // Genome Biol. 2003; 4(9): R55.

[49] Sciamanna I, Vitullo P, Curatolo A, Spadafora C. Retrotransposons, reverse transcriptase and the genesis of new genetic information // Gene. 2009; 448(2): 180-6.

[50] Spadafora C. A reverse transcriptase-dependent mechanism plays central roles in fundamental biological processes // Syst Biol Reprod Med. 2008; 54(1): 11-21.

[51] Landweber LF, Kari L. The evolution of cellular computing: nature's solution to a computational problem // Biosystems. 1999; 52(1-3): 3-13.

[52] Simon HA. The architecture of complexity // Proceedings of the American Philosophical Society. 1962; 106(6): 467-82.

## Appendix

```
(* Repressilator *)
directive sample 50000.0
directive plot !a as "a"; !b as "b"; !c as "c"
directive graph

val bind = 10.0       (* protein binding - r *)
val transcribe = 0.1 (* constitutive expression - epsilon *)
val unblock = 0.001  (* repression delay - eta *)
val degrade = 0.001  (* protein decay - delta *)

(* transcription factor *)
let tr(p:chan()) =
  do !p; tr(p)
  or delay@degrade

(* neg gate *)
let neg(a:chan(), b:chan()) =
  do ?a; delay@unblock; neg(a,b)
  or delay@transcribe; (tr(b) | neg(a,b))

(* circuit *)
new a@bind:chan()
new b@bind:chan()
new c@bind:chan()

run (neg(a,b) | neg(b,c) | neg(c,a))
```

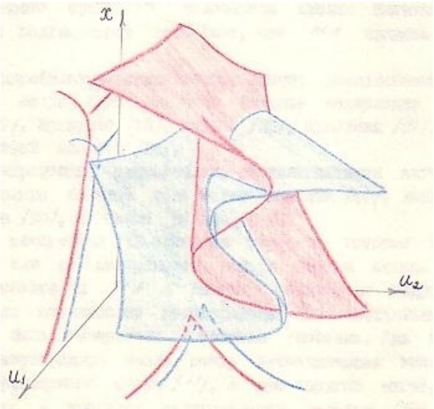


**Figure 1.** Soluble (red) and insoluble (blue) forms of GFA protein in nervous cells, where are *x* – GFA protein content, $u_1$ – malignancy, $u_2$ – putative immune response [20].

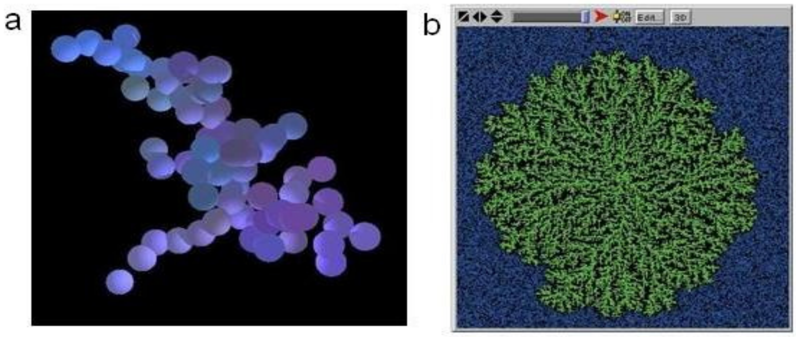


**Figure 2.** Diffusion limited aggregation (DLA), with a – 3D simulation, b – 2D simulation.

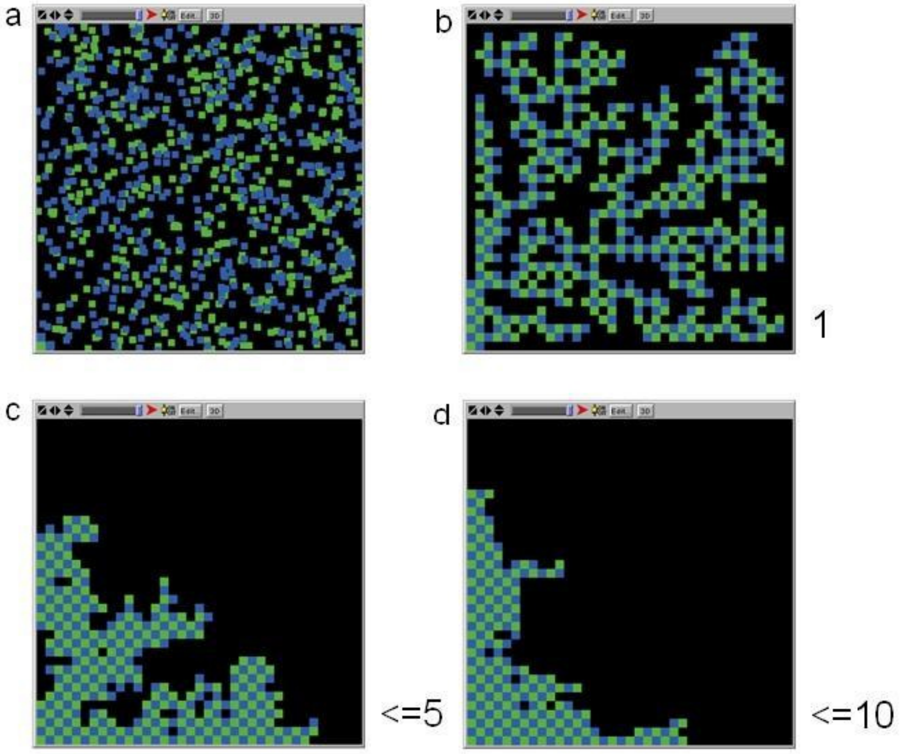


**Figure 3.** The chessboard pattern formation by adhesion rules in presence of DLA, where are a – initial state, b-d – final states with different values of diffusion.

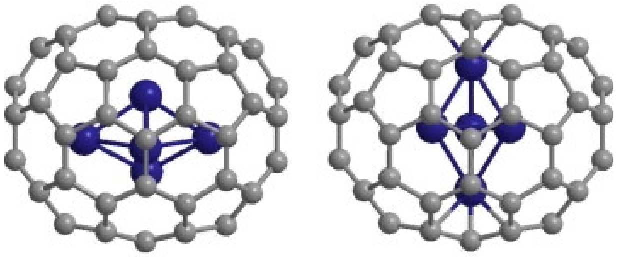
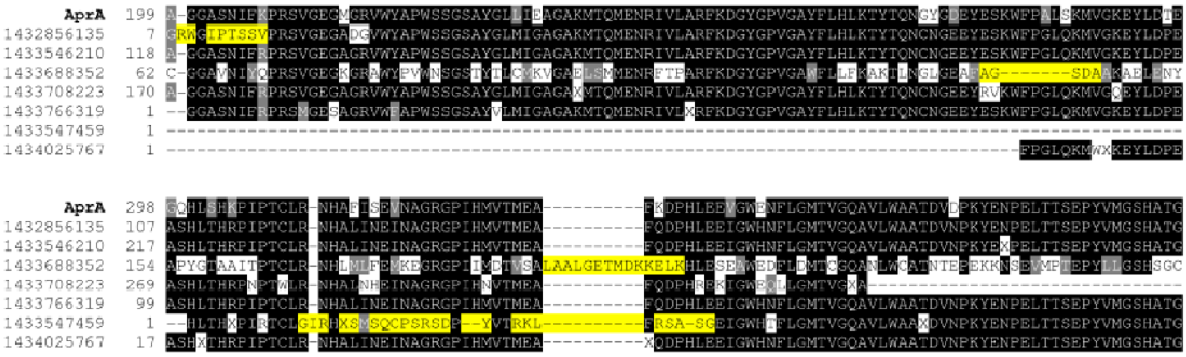


**Figure 4.** Cobalt-5 cluster within fullerene C70 in along and across orientations.



```
      AprA 199 A-GGASNIEKPRSVGEGMGRVWWYAPWSSGSAYGLLIEAGAKMTQMENRIVLARFKDGYGPVGAYFLHLKTYTQNGYGDEYESKWFPALSKMVGKEYLDTE
1432856135   7 GRWGIPTSSVPRSVGEGADGVWWYAPWSSGSAYGLMIGAGAKMTQMENRIVLARFKDGYGPVGAYFLHLKTYTQNCNGEEYESKWFPGLQKMVGKEYLDPE
1433546210 118 A-GGASNIFRPRSVGEGAGRVWWYAPWSSGSAYGLMIGAGAKMTQMENRIVLARFKDGYGPVGAYFLHLKTYTQNCNGEEYESKWFPGLQKMVGKEYLDPE
1433688352  62 C-GGAVNIYQPRSVGEGKGRAWWYPVWNSGSTYILCMKVGAELSMMENRFTEARFKDGYGPVGAWFLLFKAKTLNGLGEAFAG--------SDAAKAELENY
1433708223 170 A-GGASNIFRPRSVGEGAGRVWWYAPWSSGSAYGLMIGAGAXMTQMENRIVLARFKDGYGPVGAYFLHLKTYTQNCNGEEYRVKWFPGLQKMVGQEYLDPE
1433766319   1 --GGASNIFRPRSMGESAGRVWWFAPWSSGSAYVLMIGAGAKMTQMENRIVLXRFKDGYGPVGAYFLHLKTYTQNCNGEEYESKWFPGLQKMVGKEYLDPE
1433547459   1 -----------------------------------------------------------------------------------------------------
1434025767   1 -------------------------------------------------------------------------------------FPGLQKMWXKEYLDPE

      AprA 298 GQHLSHKPIPTCLR-NHAFISEVNAGRGPIHMVTMEA----------FKDPHLEEVGWENFLGMTVGQAVLWAATDVDPKYENPELTTSEPYVMGSHATG
1432856135 107 ASHLTHRPIPTCLR-NHALINEINAGRGPIHMVTMEA----------FQDPHLEEIGWHNFLGMTVGQAVLWAATDVNPKYENPELTTSEPYVMGSHATG
1433546210 217 ASHLTHRPIPTCLR-NHALINEINAGRGPIHMVTMEA----------FQDPHLEEIGWHNFLGMTVGQAVLWAATDVNPKYEXPELTTSEPYVMGSHATG
1433688352 154 APYGTAAITPTCLR-NHLMLFEMKEGRGPIIMDTVSALAALGETMDKKELKHLESEAWEDFLDMTCGQANLWCATNTEPEKKNSEVMPTEPYLLGSHSGC
1433708223 269 ASHLTHRPNPTWLR-NHALNHEINAGRGPIHNVTMEA----------FQDPHREKIGWEQLLGMTVGXA-------------------------------
1433766319  99 ASHLTHRPIPTCLR-NHALINEINAGRGPIHMVTMEA----------FQDPHLEEIGWHNFLGMTVGQAVLWAATDVNPKYENPELTTSEPYVMGSHATG
1433547459   1 --HLTHXPIRTCLGIRHXSMSQCPSRSDP--YVTRKL----------FRSA-SGEIGWHTFLGMTVGQAVLWAAXDVNPKYENPELTTSEPYVMGSHATG
1434025767  17 ASHXTHRPIPTCLR-NHALINEINAGRGPIHMVTMEA----------XQDPHLEEIGWHNFLGMTVGQAVLWAATDVNPKYENPELTTSEPYVMGSHATG
```

**Figure 5.** Protein alignment of the DNA reads from microorganisms living in the gutless worm *O.algarvensis* to the AprA polypeptide from bacterium *V.okutanii*, where are AprA – protein from *V.okutanii*, numbers – reads from the Gutless Worm database, yellow – sufficient insertions, deletions and exchanges.

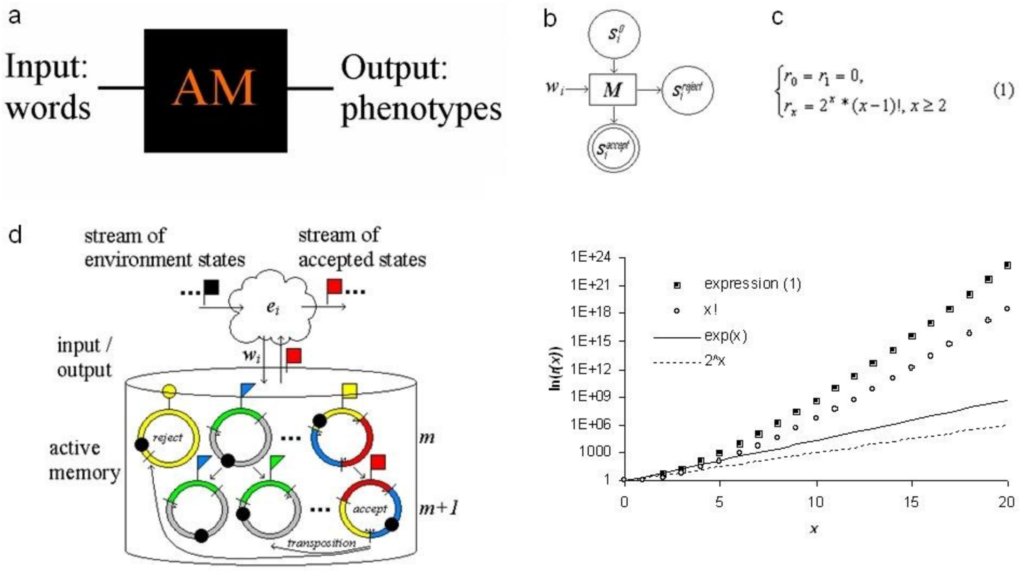


**Figure 6.** Schematics of *Argo-machine*, with a – block diagram, b – state diagram, c – combinatorial analysis, d – functional diagram.

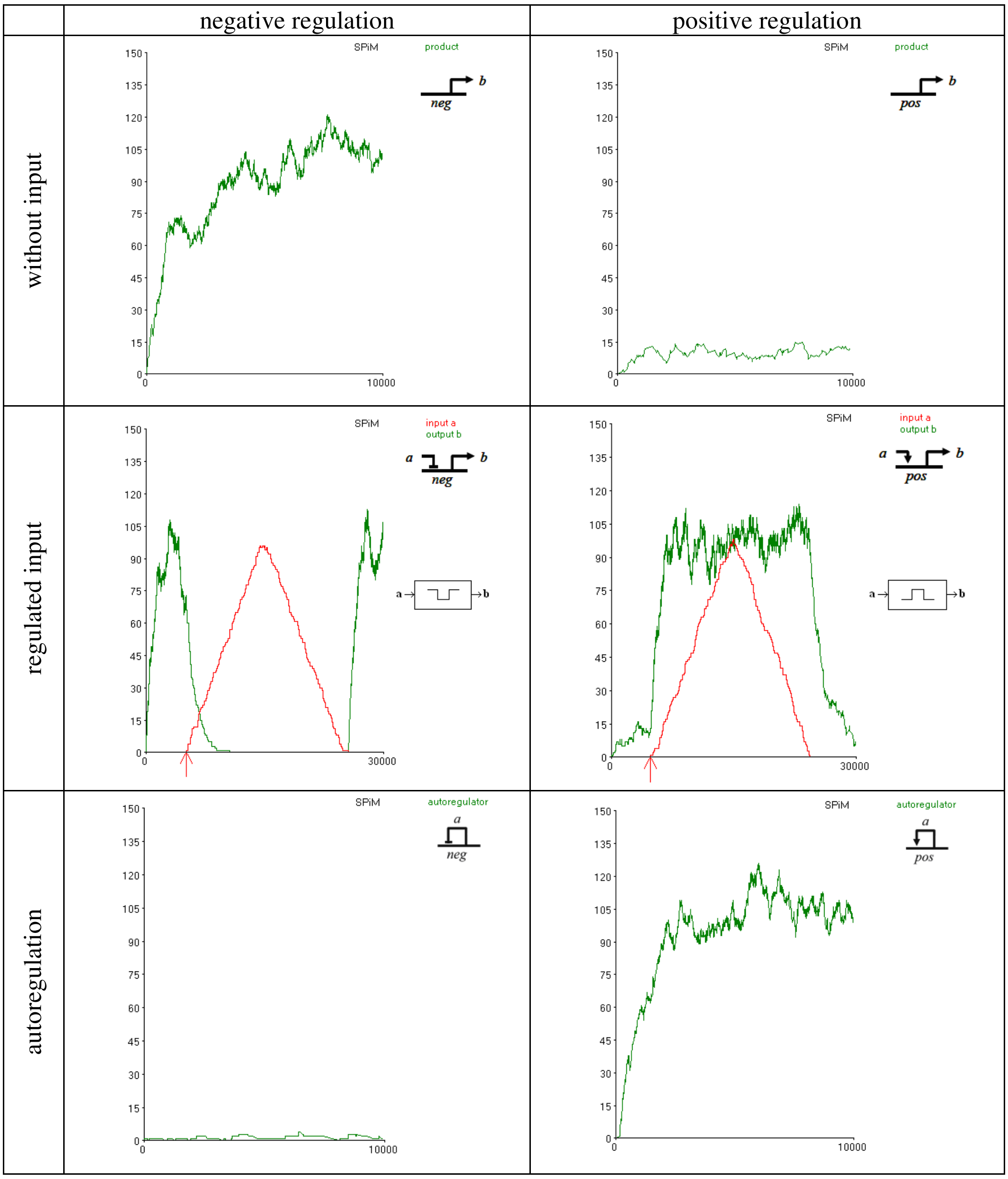


**Figure 7.** Basic genetic gates.

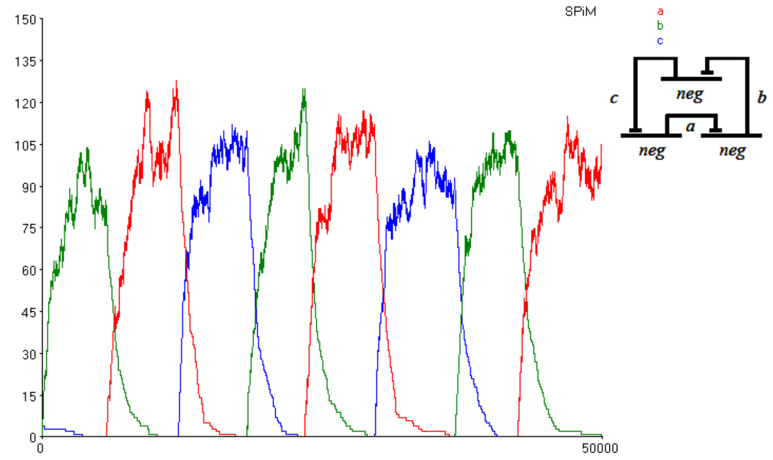


**Figure 8.** Repressilator, where r = 10.0, $\delta = 0.001$, $\varepsilon_n = 0.1$, $\eta_n = 0.001$ [36].

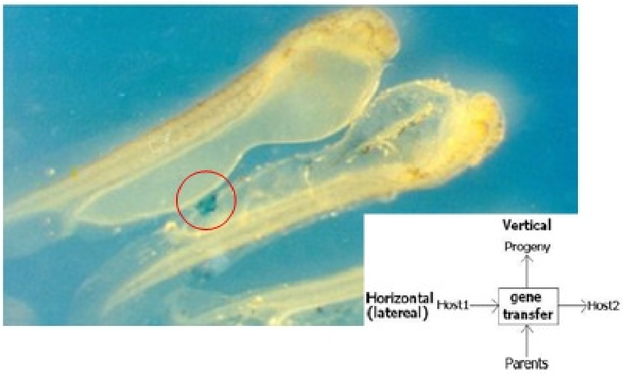
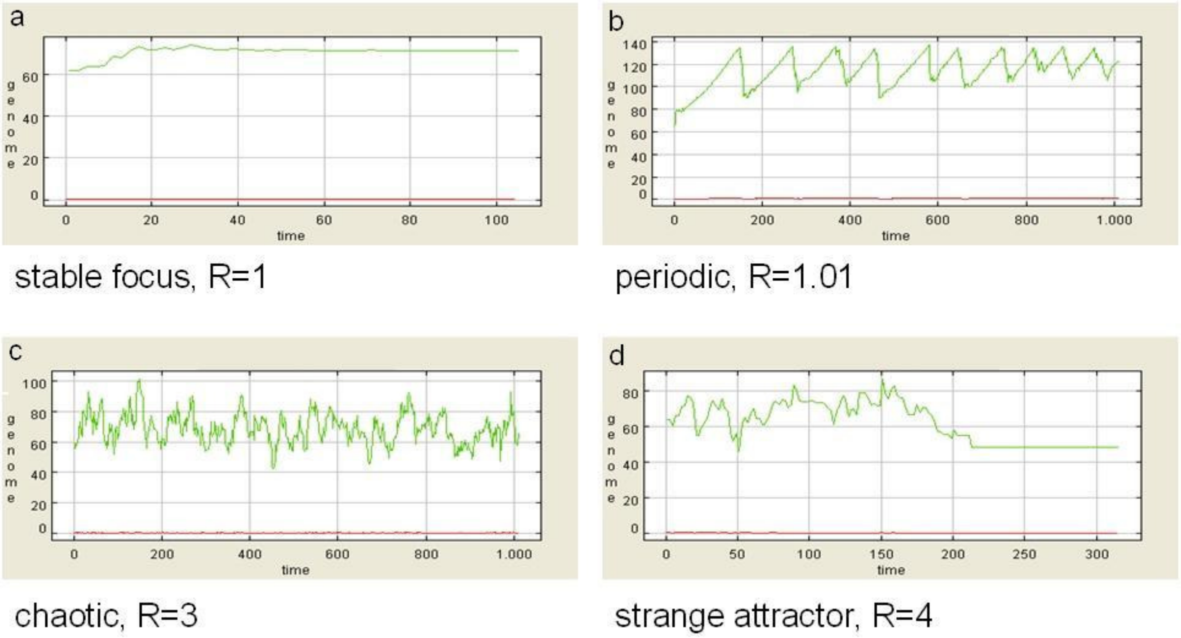


**Figure 9.** Sperm-mediated gene transfer of the plasmid pcDNA3-*lacZ* into *M.fossilis*.



stable focus, R=1

periodic, R=1.01

chaotic, R=3

strange attractor, R=4

**Figure 10.** Behavior depending on mutation parameter R, where the abscissa is a time of simulation, the ordinate is an average genome.

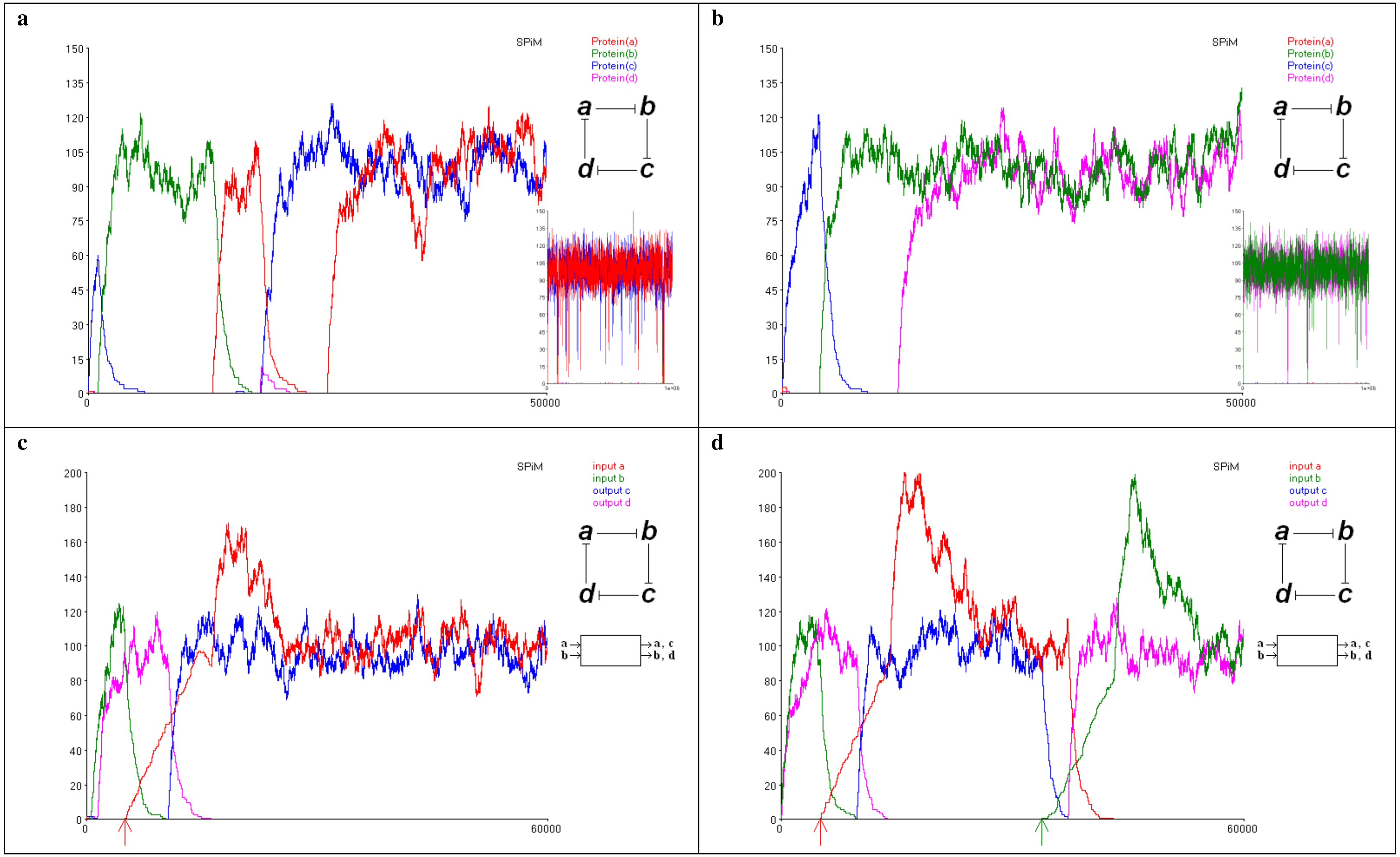


**Figure S1.** Memory, where a – initial *ac* state, b – initial *bd* state, c – input *a* switches *bd* to *ac* state, d – input *b* switches *ac* back to *bd* state.

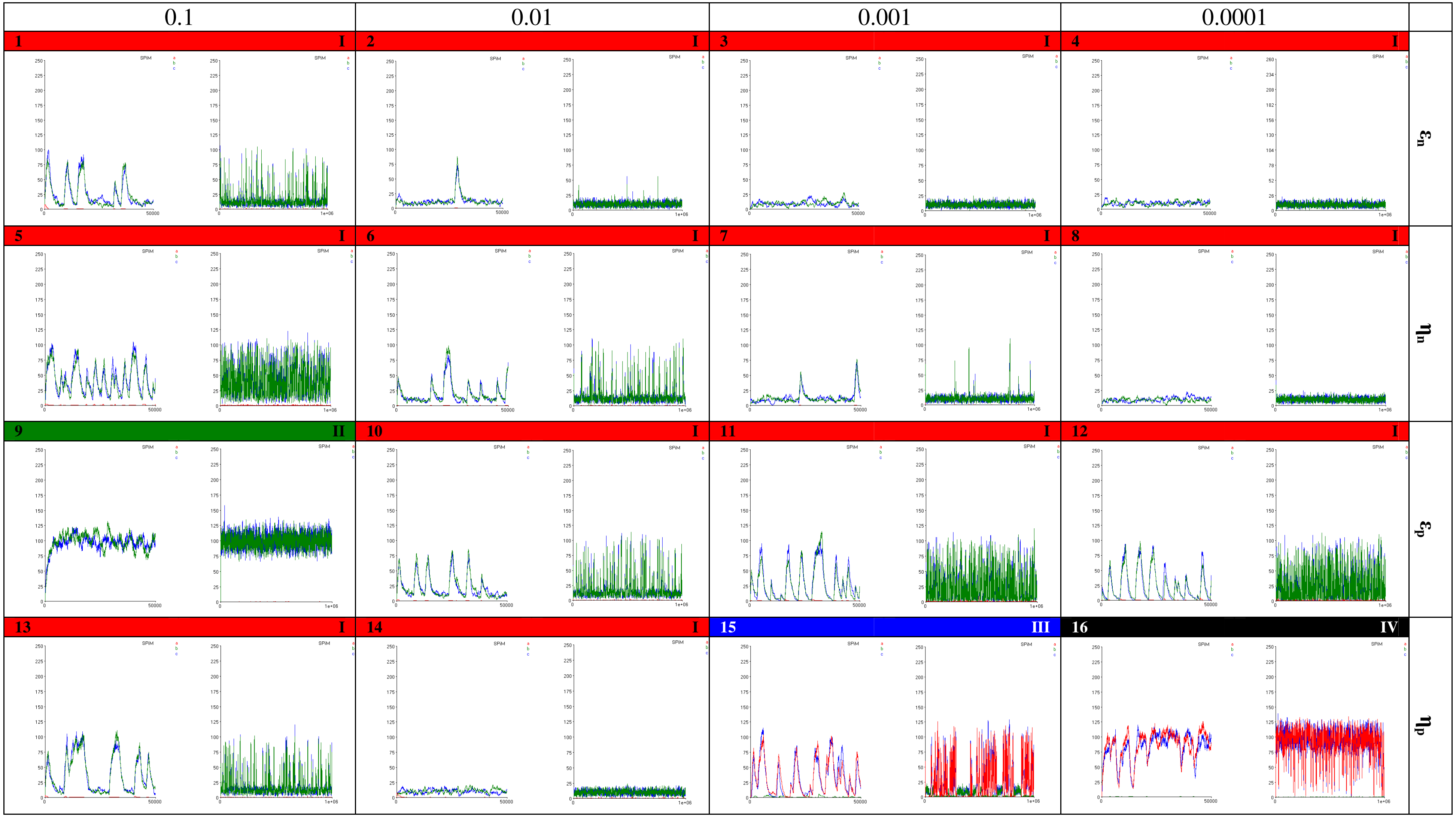


**Figure S2.** Coherent FFBL (short- and long-time simulations), see [36] for details.

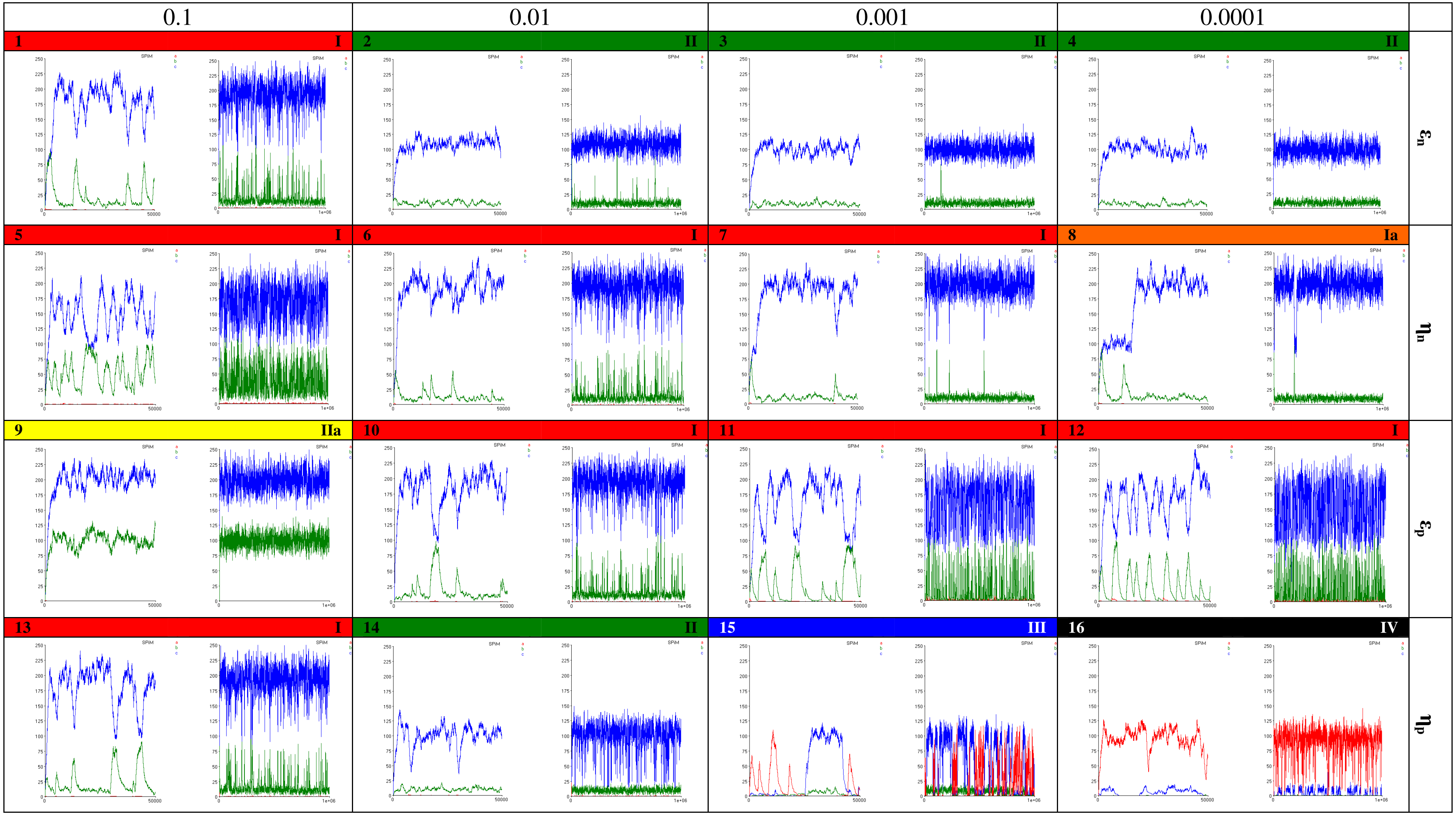


**Figure S3.** Incoherent FFBL (short- and long-time simulations), see [36] for details.