Biocybernetics and Biosemiosis

Donald Johnson

Retired Scientist and Professor (APU, U-MD, U-MN & U-WI), 38762 Fawn Springs Dr, Palm Desert, CA 92260, don@programmingoflife.info

Abstract

Biocybernetics is the study of life's hardware and software systems, which control the chemistry and physics of all of life's processes, including metabolism, manufacturing, control, and feedback. Unlike chemistry and physics, which are physical sciences, biology is an information science since what differentiates biology from complex organic chemistry is its information processing systems. Semiosis connects two independent worlds of signs and meaning by the conventional rules of a code. Many arbitrary coded symbol systems, with over 20 discovered in the past decade, play very important roles in communicating information between life's components. Life's networked computers and computer programs instantiated into DNA and RNA memory devices are discussed. A prescriptive algorithm can be implemented in either hardware or software. The "artificial genome" manufactured by Venter *et al*. demonstrates experimentally the reality of computer hardware and software in each cell.

Any serious origin-of-life or origin-of-species scenario must explain the origin of the required biological information. It is argued that each protein arises as the result of the execution of a genuine computer program. The creation of a functional protein via the mutation/selection paradigm lacks support from information science. Those who understand the reality of bioinformation, especially the prescriptive information of biocybernetics, will be able to incorporate that understanding into new models that will lead to a more complete understanding of life.

Key words: biosemiosis, biocybernetics, prescriptive information, DNA software, artificial genome, life's computers

Introduction

Biocybernetics is the study of life's hardware and software systems that use digital information processing to control, integrate, and maintain life's processes. While physics and chemistry are physical sciences whose interactions are wholly determined by physicality, biology is an information science since all of the defining characteristics of biology are controlled by life's information processing systems. Biology isn't just complicated chemistry, since it involves coded messages (semiotics) [1, 2] and coded algorithmic prescriptive instructions (instantiated computer programs) [3–5]. The vital nature of information in life has been downplayed by most materialists, since functional information has no feasible cause from physicality (though infeasible scenarios have been speculated). When addressed at all, the informational aspects of life are usually treated as metaphors or analogies, rather than realities.

Information is a non-material entity that can be instantiated into physicality for storage or communication. Information always involves contingency, such that it could have been different. If there is no contingency, the value is not informational, but instead is determined by natural law. Any property determined wholly by law is not informational. It is common to mistakenly view a physical property of an object as information. For example, the temperature of an object is a property totally determined by a number of physical laws related to mass, specific heat, energy flow, radiation, etc. Since temperature is determined by law, that property is not information, even though it can be transduced into functional information by use of a device known as a thermometer (which could be part of a thermostat for controlling the temperature). The temperature of an object could be information if it were contingent through appropriate choice of constraints. For example, a rock could convey binary information: hot = yes, cold = no, where the information rock is placed in either a bed of coals or a glacier to record the choice, before placing it in an insulated container for later examination. Obviously this stored information would be lost with time, as the rock nears ambient temperature, but a bit of RAM memory in your computer also requires a refresh to retain its information. The radiation from a star is totally determined by physicality, but a spectrophotometer could be used to produce information related to the star's temperature, composition, velocity, etc. The star's radiation has no contingency in that its properties cannot be otherwise give the initial constraints and the laws involved. The human measurements of that radiation, on the other hand, involve considerable contingency, and could even be incorrect if the instruments weren't properly calibrated. The weather is totally determined by law and initial constraints. Even a dark cloud with rain descending is not informational without an observer with the capability to ascertain meaning from the physical properties observed.

The broadest classification of information is that from information theory developed by Shannon [6], which requires nothing meaningful, except in the case of a coded information subset. Uncertainty is a better descriptive term since the Shannon "Information" of a pattern is inversely related to the pattern's probability. A random sequence has the highest possible uncertainty. A subset of the broadest category is functional information, which has meaning (such as in coded information). The most restrictive classification of information is prescriptive information, which is not only functional/meaningful, but is algorithmic (a recipe). Consider data typed into a word-processing program. Most such data is functional in that it has a purpose of communicating information to the ultimate reader of that information. If a monkey typed random data into the program, that complex data, produced by chance contingency, would have no purpose, but would have a very high Shannon uncertainty since that deals only with the probability of the data pattern, irrespective of any meaning. A computer program typed into the word processor is more than just meaningful, but is prescriptive in that it contains instructions to accomplish objectives based on data to be supplied during the execution of the program being typed. Prescriptive information may be a simple step-wise recipe, or may express the decisions to be made and the criteria for the different execution paths.

Some have questioned whether there are actual computer programs (instantiated algorithms) in life, or there is just a "resemblance" to computer-like characteristics. One of the most significant experimental confirmations of the reality that life is hardware/software was the announcement in 2010 of Venter's computer-generated artificial genome [7]. Venter stated "It certainly changed my views of definitions of life and how life works... Life is basically the result of an information process, a software process. Our genetic code is our software" [8]. Venter's team didn't "create life," but they put life synthesized pieces of the target DNA into yeast which assembled the target bacterium's genome. They didn't engineer specific instructions (algorithms), but rather combined DNA blocks that matched the target sequence. The assembled genome was transplanted into a different bacterium and 'booted up' to create a new synthetic version of the target. For this "proof of principle" instance, they synthesized a bacterium as close to the original genome as they could, using the original DNA as a "standard" for comparison, replacing the genome's application program set stored in the original organism's DNA memory with a genetically engineered application program set matching the target. The operating systems and the interacting computers in the cell whose genome was replaced remained intact and were able to function by using the replacement software. One of the things this research demonstrated is that (at least for the two bacteria involved) life uses common operating systems, programming languages, and devices (otherwise the programs for one machine wouldn't execute on another).

Since many believe it is important to differentiate hardware from software, perhaps it is beneficial to consider some related computer science principles. To be functional, both hardware and software are instantiations of algorithms, which are step-by-step solutions to problems. In the case of hardware, the algorithm could be developed using state-transition diagrams or a hardware description language before instantiation in an electronic circuit. Any hardware-generated control signals could be generated by software control. There is also an important distinction in computer science between architecture and organization.

Computer architecture refers to the machine characteristics visible to lowest-level user programming (the assembly language instruction set). Organization refers to the implementation of the architecture. For example, a CPU's control unit uses the fetched machine language instruction, along with other inputs, to generate the control signals needed to carry out the instruction. A control unit could be purely electronic, which is often done for the fastest computers. A control unit is usually implemented using a less-expensive control storage interpreter which uses the machine instruction to generate an address for reading the instruction's control signals from a control memory (microcode). The control storage can be permanent or writeable (allowing different machine architectures using the same hardware). The functionality is identical for any organization for the same architecture, and organization couldn't be ascertained from functionality. This is important for life's information because it may not be critical to identify what is software and what is hardware when analyzing functionality. Since hardware and software can't be differentiated based on functionality in electronic computers, there is no information science reason to expect differentiation would be possible based on functionality of biocybernetic systems. That differentiation may be important when ascertaining mechanisms, however.

Life's Computers

Most people tend to have a very narrow view as to what a computer can be. Realize that the first computer, Babbage's 1837 Analytic Engine (Fig.1), was totally mechanical, and yet "Turing complete" (could theoretically be programmed to compute anything possible to compute). Many architectures and organizations can be classified as "computers" since the necessary and sufficient requirements for a computer are: input (or embedded data), memory, an instantiated program, processing capability, and output. Note that the first electronic computer was not Turing complete (no branching capability) and couldn't be re-programmed, so those characteristics (as found in many biological computers) aren't required to be a "computer." There are many components of life that can thus be classified as computers or components of computers, so that the reality and variety of biological computers should not be surprising. For example, multiple proteins (including RNA polymerase) may form a computer to read the DNA memory/program to output the mRNA transcription. The "program" could be in the non-coding DNA, which could use the "gene" as data to transcribe. Some hypothesize that the transcription components are "merely" under control of a master computer, and are equivalent to a disk head assembly (perhaps with built-in control as found in a hard drive's read/write head assembly) [9]. Without being dogmatic,

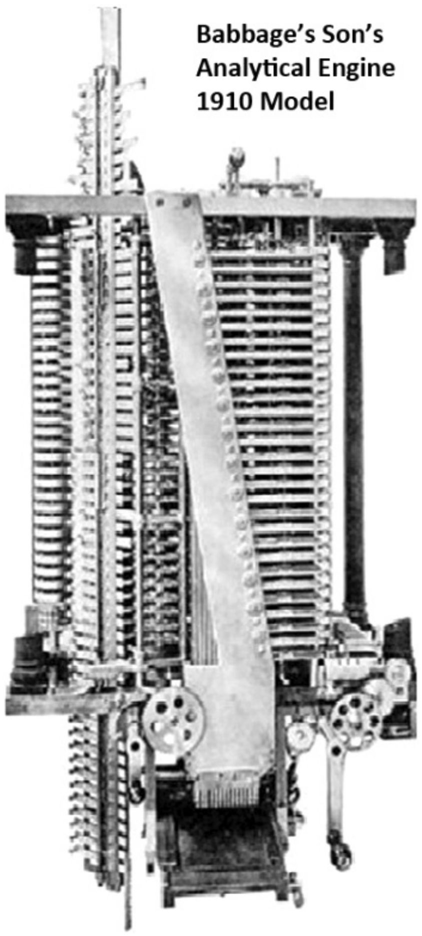

Fig. 1. Babbage Mechanical Computer – Babbage's 1837 Analytic Engine, was totally mechanical, and yet "Turing complete".

that alternative approach wouldn't explain the fact that the replisome has higher priority than polymerase, causing a transcription in progress to abort [10] (why would a "master" control computer start such a transcription?). High-performance pipelined computers often use "optimistic scheduling" to start operations that won't ultimately complete, but this would seem to be a waste of energy for a process of life. Multiple networked interacting semi-autonomous computers seem to fit the observations better (at least based on what is currently known). Is the mRNA "simply" a coded digital message for the ribosome to process, or is mRNA a program to be interpreted by the ribosome? The later seems likely since mRNA can be generated by multiple means, each producing a protein as the output during the execution of the computationally-halting program (a requirement of a functional algorithm) when the mRNA program is interpreted by the ribosome (equivalent to a micro-programmed control unit interpreting a machine instruction sequence). Since the ribosome contains RNA memory, in addition to a multitude of proteins, and interprets the prescriptive program of its input mRNA, it seems likely that a ribosome is indeed a genuine specific-purpose computer (it has all necessary and sufficient requirements). It should also be noted that the epigenome, polypeptides (including proteins) and micro-RNAs of various lengths can serve as information-carrying structures and/or memories. The author has peer reviewed publications using serially-shared information within multiprocessor systems [11–13], and can attest to the importance of protocols for functionality when communicating such information.

When examining tRNA, a computer scientist quite naturally considers the purpose of its RNA memory structure. Does tRNA operate totally by "law," or might this be another computer? Although the total mechanism for attaching a particular amino acid so that it matches the codon on the opposite end of the tRNA is complex, and not fully understood, the presence of RNA, a memory structure, may indicate that multiple proteins can form a computer with the tRNA's instruction memory to select and attach the appropriate amino acid, and release it as output at the ribosome's request. Once again, the tRNA complex possesses the necessary and sufficient characteristics that define a computer. If a mechanism based on law can explain the functionality of tRNA, then perhaps its RNA memory simply serves as a separator, as opposed to being functional memory (which seems unlikely to the author). In any case, each protein is the result of the execution of a real computer program, ultimately instantiated in DNA for the protein's sequence. Venter's artificial genome experiment demonstrated that even mRNA generated by alternative mechanisms than direct transcription ultimately depends on the DNA memory.

Thus far, there has been no feasible mechanism proposed for writing computer programs by inanimate nature. There also has been no feasible mechanism for computer hardware being implemented from inanimacy. All known computer programs and hardware systems require formal solutions before a functional result is obtained. The prescriptive information incorporated in both life's hardware and software currently lacks any feasible explanation from chance and necessity. Scientific answers are needed, as no scenarios proposed so far are compatible with information science. The Origin-of-Life Prize (www.lifeorigin.info) highlights the major difficulties and "will be awarded for proposing a highly plausible mechanism for the spontaneous rise of genetic instructions in nature sufficient to give rise to life" [14]. OOL requires that each nucleotide of the RNA sequence be selected for potential function, as opposed to natural selection's favoring of existing functionality. Since natural selection depends on already existing protein structures of the phenotype, and each protein is a result of the genomic algorithm instantiated in the DNA, natural selection is not a mechanism for generation of new prescriptive information, but at most is a sorting procedure to weed out organisms that are less fit. What mutation/selection really says is that, "randomly changing a functional program can sometimes produce a modified program with improved functionality." Such a random net increase in non-trivial functionality has never been documented in computer science. For example, random changes in so-called "artificial life" programs use designed targets and fitness functions to steer results in desirable directions for functionality [15,16]. When irreducibly complex structures are considered, multiple programs would require simultaneous modification.

The polynucleotide sequence of DNA or RNA is an ideal information storage structure since each nucleotide has no dependence on preceding or following nucleotides, and can be arbitrarily set to the functional value desired from the four possible values. It should also be mentioned that within the DNA helix, only half of the nucleotides are informational since one strand is totally determined (and is redundant) by the other (informational) strand. Information requires contingency, and one strand has none. Note that other information for decoding overlapping genes and reverse transcription is not directly in the DNA sequence. The prescriptive information in a DNA sequence is a recipe or algorithm to accomplish a desired task. What complicates this is the fact that many nucleotides are components of multiple prescriptions, such as in overlapping genes or alternative splicing schemes. In those cases, the nucleotide has to be set so that it becomes a functional component of multiple algorithms. The algorithms can be those for protein generation or one of the numerous cellular controls. Sub-coded (codes within codes) information [17] and a second genetic code [18] characterizing alternative splicing have been discovered. Various transcribed RNAs are mixed and matched and spliced into mRNAs for specifying protein construction and other controls, sometimes joining messages that were separated by thousands of nucleotides. "For example, three neurexin genes can generate over 3,000 genetic messages that help control the wiring of the brain" [19]. Even "simple" prescription information lacks any feasible explanation using known science. Much more challenging are the explanations required for multiple and overlapping levels of prescriptive information.

Biosemiosis

"Physicality is the only reality" is a paradigm which encounters severe difficulties when confronted with biological coding systems (semiosis), and the associated formal operations which are required. What is a semiotic system? “A semiotic system is a system made of two independent worlds that are connected by the conventional rules of a code. A semiotic system ... is necessarily made of three distinct entities: signs, meanings and code” [20]. The best-known biological code is the codon-to-amino acid translation during protein construction which uses tRNAs to translate one codon from the 64-codon alphabet (a sign) into one amino acid in the 20 amino acid alphabet (meaning). There is no chemical or other deterministic link between the opposite ends of a tRNA that causes a particular amino acid to be associated with a particular codon. They are associated by an arbitrary rule determined by a code. Over 20 other semiotic codes have been discovered in life in the past decade, with each code having arbitrary rules agreed on by both sender and receiver of the coded information message, as described briefly below.

A coactivator code for coregulators may confer specificity to ubiquitous transcriptional regulatory factors, with wide-reaching implications. Cofactors use a variety of mechanisms to contribute to gene transcription activation and repression [21]. A protein destination code ensures delivery of the protein to the correct destination. “Proteins are the workhorses of the cell, but to get the most work out of them, they need to be in the right place. In neurons, for example, proteins needed at axons differ from those needed at dendrites, while in budding yeast cells, the daughter cell needs proteins the mother cell does not. In each case, one strategy for making sure a protein gets where it belongs is to shuttle its messenger RNA to the right spot before translating it. The destination for such an mRNA is encoded in a set of so-called “zipcode” elements, which loop out of the RNA string to link up with RNA-binding proteins. In yeast, these proteins join up with a myosin motor that taxis the complex to the encoded location” [22]. A code for resolving overlapping codes is needed to start transcription appropriately. “Genomes encode multiple signals, raising the question of how these different codes are organized along the linear genome sequence” [23]. The detailed coding “signals consist of both known and potentially novel codes, including position dependent secondary RNA structure, bacteria-specific depletion of transcription and translation initiation signals, and eukaryote-specific enrichment of microRNA target sites” [23].

The cytoskeleton anchoring code [1] determines the ultimate relationship between the cellular structures that the cytoskeleton is working on and the microtubule and microfilament components of the cytoskeleton. Every microtubule starts from a centrosome, with the other end growing or contracting in an exploratory “strategy” in a search for an anchor. There is a “dynamic instability” as monomers are added and taken away (if an anchor is not found), so the cytoskeleton can rapidly explore all of the cytoplasm’s space, until a stable anchor code is found.

Barbieri lists 20 semiotic codes [20] from a variety of research papers, including adhesive code, sugar code, histone code, neural transcriptional codes, regulatory code in mammalian organogenesis, code of post translational modifications, nuclear receptors combinatorial code, transcription factors code, acetylation codes, estrogen receptor code, metabolic codes, rna codes, error-correcting codes, modular code of the cytoskeleton, lipid-based code in nuclear signaling, immune self-code, and signal transduction codes. In each case, the code provides an arbitrary translation between disjoint domains. This list only scratches the surface of all the codes that are still waiting to be discovered.

Since information is non-material, there have been no feasible scenarios for production of semiotic systems from physicality. Barbieri proposes "natural conventions" as the required codemaker that creates the required semiotic translation bridge between the sender and receiver [20]. Barbieri fails to present a feasible mechanism, but argues it "must have happened" since semiosis is a ubiquitous reality and is actually the mechanism he proposes for macroevolution. In his view, for something totally new to appear, a new organic code that had never existed before must come into being. Biological specificity (required for heredity and reproduction) was the result of the origin of the genetic code. Signal transduction codes allowed systems to produce their own signals, separating their internal space from the environment. The origin of the eukaryote nucleus was brought about by the origin of splicing codes [1,24]. The development of any coding system must account for information (especially transfer of information), in a manner compatible with information theory. The next paragraph provides the technical details, but the bottom line is that codes cannot evolve from simpler to more complex basic codes without violating an information theory theorem that has stood for over 60 years.

Given the probability vector, $\mathbf{p_A}$, of the elements of alphabet A in a source probability space [Ω, A, $\mathbf{p_A}$] and the probability vector, $\mathbf{p_B}$, of the elements of alphabet B in destination probability space [Ω, B, $\mathbf{p_B}$], then a unique mapping of the symbols of alphabet A onto the symbols of alphabet B is called a code [25]. Mutual entropy is a mathematical measure of the similarity between any two sequences one wishes to compare. Mutual entropy relates the input (x) and output (y) channels via: $\mathbf{I(B;A) = I(A;B) = H(x) - H(x|y)}$, where the conditional (x_i given y_i received) entropy is $\mathbf{H(x|y) = -\sum_{ij} p_j\, p(i|j) \log_2 p(i|j)}$, $\mathbf{p_j = \sum_i^n p_i p(j|i)}$ (which relates the probability vector, **p**, elements to those of the conditional probability matrix, **P**), and $\mathbf{H(x) = -\sum_{i=1}^{n} p_i \log_2(p_i)}$ is the information entropy. The Shannon Channel Capacity is also the maximum mutual entropy. For a transmitting system with fewer symbols in [Ω, *A*, $\mathbf{p_A}$] to pass information to [Ω, *B*, $\mathbf{p_B}$], the maximum mutual entropy would be exceeded. The channel capacity thus prohibits a simpler symbolic alphabet (e.g. a 2-nucleotide "codon") from evolving into an alphabet with more intrinsic symbols. Some have suggested it "must have happened," but have provided no falsification of Shannon Channel Capacity Theorem that has stood for over 60 years. Without such falsification, the original instantiation of any semiotic code would have an alphabet at least as symbolically complex as that currently used. Note that an alphabet symbol may consist of other symbols. For example, the American Standard Code for Information Interchange (ASCII) defines printable characters represented by seven bits, with a 7-bit group being one sign for the source alphabet.

The growing acknowledgment that the mutation/selection model of evolution is not sufficient to explain the origin of elaborate information processing systems seems to suggest that a major paradigm shift is imminent. The leading contender as a replacement is the "Extended Synthesis" [26], which is very flexible, incorporating essentially all proposed mechanisms for evolution, including concepts like "natural genetic engineering" [27, 28] for mutation selection and "natural conventions" [20] as semiotic code. Whatever the replacement will be, science needs to ensure that any scenarios are compatible with information science. The most difficult realities to accommodate will probably be the prescriptive information of biocybernetics and the arbitrary information translation codes of biosemiosis. There are numerous very specific problems that must explained if the the neo-Darwinian paradigm is to survive. Given all these challenges the defenders of the status quo must provide scientific answers to a series of extremely difficult questions, include the following [14].

1. *How did nature write the prescriptive programs needed to organize life-sustaining metabolism? Programs are shown by computer science to require a formal solution prior to implementation. How did inanimate nature formally solve these complex problems and write the programs? How did nature develop the operating systems and programming languages to implement the algorithms? How did nature develop Turing machines capable of computational halting? How did nature develop the arbitrary protocols for communication and coordination among the thousands (or millions) of computers in each cell?*

2. *How did nature develop multiple semiotic coding systems, including the redundant (surjective) codon-based coding system (for symbolic translation) that involves transcribing, communicating, and translating the symbolic triplet nucleotide block-codes into amino acids of the proteins? How did nature develop alternative generation of such messages using techniques such as overlapping genes, messages within messages, multi-level encryption, and consolidation of dispersed messages? A protein may obtain its consolidated prescriptive construction instructions from multiple genes and/or from the "junk" DNA, sometimes with over a million nucleotides separating the instructions to be combined.*

3. *How did nature defy computer science principles by avoiding software engineering's top-down approach required for complex programming systems? How did nature produce complex functional programs without planning, by randomly*

modifying existing algorithms? How did multiple such programs become simultaneously modified to result in the production of irreducibly complex structures?

These questions demand scientific answers that are compatible with information science. "It must have happened" is not science, but belief.

References

1. Barbieri M (2003) The Organic Codes. An introduction to semantic biology. Cambridge University Press, Cambridge, UK.
2. Barbieri M (editor of anthology) (2008) The Codes of Life: The Rules of Macroevolution.
3. Abel D (2008) The 'Cybernetic Cut': Progressing from Description to Prescription in Systems Theory. The Open Cybernetics and Systemics Journal 2: p252–262.
4. Abel D (2009) The GS (genetic selection) Principle. Frontiers in Bioscience 14: 2959–2969.
5. Abel D (2009) The Biosemiosis of Prescriptive Information. Semiotica, 1/4/09, p1–19.
6. Shannon (1948) A Mathematical Theory of Communication. Bell System Technical Journal: 27, July & October, p379–423 & 623–656.
7. Gibson D, et al (2010) Creation of a Bacterial Cell Controlled by a Chemically Synthesized Genome. Science 329(5987):52–56.
8. Venter (Craig), Interview, http://www.guardian.co.uk/science/video/2010/may/20/craig-venter-new-life-form
9. D'Onofrio D, An G (2010) A Comparative Approach for the Investigation of Biological Information Processing: an Examination of the Structure and Function of Computer Hard Drives and DNA. Theoretical Biology and Medical Modeling 7:3.
10. Rockefeller University (2010) Scientists crash test DNA's replication machinery. ScienceDaily, 3/1/10.
11. Johnson D, Lilja D, Riedl J, "A Circulating Active Barrier Synchronization Mechanism," International Conference on Parallel Processing I, 8/95, p202–209.
12. Johnson D, Lilja D, Riedl J, Anderson J, "Low-Cost, High-Performance Barrier Synchronization on Networks of Workstations," Jour Par & Distributed Computing, 2/97, p131–137.
13. Johnson D, Lilja D, Riedl J (2005) Circulating Shared-Registers for Multiprocessor Systems. Jour. of Systems Arch., 6/3/05, p152–168.
14. Johnson D (2010) Programming of Life, 2010, p83–84.
15. Truman R, "Evaluation of Neo-Darwinian Theory Using the Avida Platform," PCID 3.1.1,11/04.

16. Gilder G (2006). Evolution and Me. National Review, 7/17/06.
17. Cannarozzi (G.), N. Schraudolph, M. Faty, P. von Rohr, M. Friberg, A. Roth, P. Gonnet, G. Gonnet, & Y. Barral, "Role for Codon Order in Translation Dynamics," Cell 141, 4/16/10, p355–367.
18. Barash Y, Calarco J, Gao W, Pan Q, Wang X, Shai O, Blencowe B, B, Frey B (2010) Deciphering the Splicing Code. Nature 465 (7294):53–9.
19. Frey (J.), quoted in "Researchers Crack 'Splicing Code,' Solve a Mystery Underlying Biological Complexity," 5/5/10, www.physorg.com/news192282850.html
20. Barbieri M (2008) Biosemiotics: a new understanding of life. Naturwissenschaften 95:577–599.
21. Gamble M, Freedman L (2002) A coactivator code for transcription. TRENDS in Biochemical Sciences 27 (4):165–167.
22. Robinson R (2011) A two-step process gets mRNA loaded and ready to go. Public Library of Science: Biology, 9(4): e1001047. doi:10.1371/journal.pbio.1001047.
23. Itzkovitz S, Hodis E, Segal E (2010) Overlapping codes within protein-coding sequences. Genome Res 20:1582–1589.
24. Barbieri M (1998) The Organic Codes. The Basic Mechanism of Macroevolution," Rivista di Biologia-Biology Forum 91:481–514.
25. Yockey H (2005) Information Theory, Evolution, and the Origin of Life. Cambridge University Press, Cambridge.
26. Pigliucci M, Müller G (2010) Elements of an extended evolutionary synthesis. In: Pigliucci M, Müller G (eds) Evolution—the Extended Synthesis, The MIT Press, Cambridge, MA.
27. Witzany (Günther), Natural Genetic Engineering and Natural Genome Editing (Proceedings), 12/09, 276 pgs.
28. Shapiro J (2010) Mobile DNA and Evolution in the 21st Century. Mobile DNA 1:4.