# A Phylogenetic, Ontogenetic, and Epigenetic View of Bio-Inspired Hardware Systems

Moshe Sipper, *Member, IEEE*, Eduardo Sanchez, *Member, IEEE*,
Daniel Mange, *Member, IEEE*, Marco Tomassini, Andrés Pérez-Uribe, and André Stauffer, *Member, IEEE*

***Abstract*— If one considers life on Earth since its very beginning, three levels of organization can be distinguished: the *phylogenetic* level concerns the temporal evolution of the genetic programs within individuals and species, the *ontogenetic* level concerns the developmental process of a single multicellular organism, and the *epigenetic* level concerns the learning processes during an individual organism's lifetime. In analogy to nature, the space of *bio-inspired* hardware systems can be partitioned along these three axes, phylogeny, ontogeny, and epigenesis, giving rise to the *POE* model. This paper is an exposition and examination of bio-inspired systems within the POE framework, with our goals being: 1) to present an overview of current-day research, 2) to demonstrate that the POE model can be used to classify bio-inspired systems, and 3) to identify possible directions for future research, derived from a POE outlook. We first discuss each of the three axes separately, considering the systems created to date and plotting directions for continued progress along the axis in question. We end our exposition by a discussion of possible research directions, involving the construction of bio-inspired systems that are situated along two, and ultimately all three axes. This could give rise to novel systems endowed with evolutionary, reproductive, regenerative, and learning capabilities.**

***Index Terms*— Embryonics, epigenesis, evolutionary computation, evolvable hardware, immune systems, neural networks, ontogeny, phylogeny.**

## I. Introduction: Biological Inspiration as a Bridge from the Natural Sciences to Engineering

TRADITIONALLY, the development of the engineering disciplines (civil, electrical, computer engineering, etc.) and that of the natural sciences (physics, chemistry, biology, etc.) have proceeded along separate tracks. The natural scientist is a *detective*: faced with the mysteries of nature, such as meteorological phenomena, chemical reactions, and the development of living organisms, he or she seeks to *analyze* existing processes, to *explain* their operation, to *model* them, and to *predict* their future behavior. The engineer, on the other hand, is a *builder*: faced with social and economic needs, he or she tries to *create* artificial systems (bridges, cars, electronic devices) based on a set of *specifications* (a description) and a set of *primitives* (elementary components such as bricks, beams, wires, motors, and transistors).

These two major branches of human endeavor have been drawing closer together during the past decades. It is common nowadays for scientists to use tools created by engineers. To cite one example of many, we are witness to the systematic use of electronics in the medical world for such tasks as decoding the human genome, visually representing highly complex chemical molecules, computerized tomography, and so on.

More recently, engineers have been allured by certain natural processes, giving birth to such domains as artificial neural networks and evolutionary computation. Living organisms are complex systems exhibiting a range of desirable characteristics, such as evolution, adaptation, and fault tolerance, that have proved difficult to realize using traditional engineering methodologies. Such systems are characterized by a genetic program, the genome, that guides their development, their functioning, and their death. If one considers life on Earth since its very beginning, then the following three levels of organization can be distinguished [1], [2].

- *Phylogeny:* The first level concerns the temporal evolution of the genetic program, the hallmark of which is the evolution of species, or *phylogeny*. The multiplication of living organisms is based upon the reproduction of the program, subject to an extremely low error rate at the individual level, so as to ensure that the identity of the offspring remains practically unchanged. Mutation (asexual reproduction) or mutation along with recombination (sexual reproduction) give rise to the emergence of new organisms. The phylogenetic mechanisms are fundamentally nondeterministic, with the mutation and recombination rate providing a major source of diversity. This diversity is indispensable for the survival of living species, for their continuous adaptation to a changing environment, and for the appearance of new species.
- *Ontogeny:* Upon the appearance of multicellular organisms, a second level of biological organization manifests itself. The successive divisions of the mother cell, the zygote, with each newly formed cell possessing a copy

Manuscript received November 18, 1996; revised January 17, 1997. This work was supported in part by Grants 20-42270.94 and 21-45630.95 from the Swiss National Science Foundation.

M. Sipper, E. Sanchez, D. Mange, A. Pérez-Uribe, and A. Stauffer are with the Logic Systems Laboratory, Swiss Federal Institute of Technology, IN-Ecublens, CH-1015 Lausanne, Switzerland (e-mail: sipper@di.epfl.ch; sanchez@di.epfl.ch; mange@di.epfl.ch; andres.perez@di.epfl.ch; stauffer@di.epfl.ch).

M. Tomassini is with the Logic Systems Laboratory, Swiss Federal Institute of Technology, IN-Ecublens, CH-1015 Lausanne, Switzerland, and the Computer Science Institute, University of Lausanne, Lausanne, Switzerland (e-mail: tomassini@di.epfl.ch).

Publisher Item Identifier S 1089-778X(97)03302-X.

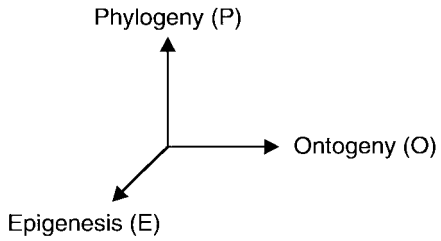


Fig. 1. The POE model. Partitioning the space of bio-inspired hardware systems along three axes: phylogeny, ontogeny, and epigenesis. See the text for definition of these terms.

of the original genome, is followed by a specialization of the daughter cells in accordance with their surroundings, i.e., their position within the ensemble. This latter phase is known as cellular differentiation. Ontogeny is thus the developmental process of a multicellular organism. This process is essentially deterministic: an error in a single base within the genome can provoke an ontogenetic sequence which results in notable, possibly lethal, malformations.

– *Epigenesis:* The ontogenetic program is limited in the amount of information that can be stored, thereby rendering the complete specification of the organism impossible. A well-known example is that of the human brain with some $10^{10}$ neurons and $10^{14}$ connections, far too large a number to be completely specified in the four-character genome of length approximately $3 \times 10^9$. Therefore, upon reaching a certain level of complexity, there must emerge a different process that permits the individual to integrate the vast quantity of interactions with the outside world. This process is known as epigenesis and primarily includes the nervous system, the immune system, and the endocrine system. These systems are characterized by the possession of a basic structure that is entirely defined by the genome (the innate part), which is then subjected to modification through lifetime interactions of the individual with the environment (the acquired part). The epigenetic processes can be loosely grouped under the heading of *learning* systems.

In analogy to nature, the space of bio-inspired hardware systems can be partitioned along these three axes: phylogeny, ontogeny, and epigenesis; we refer to this as the POE model (Fig. 1) [3]. The distinction between the axes cannot be easily drawn where nature is concerned; indeed the definitions themselves may be subject to discussion. We therefore define each of the above axes within the framework of the POE model as follows: the phylogenetic axis involves *evolution*, the ontogenetic axis involves the *development* of a single individual from its own genetic material, essentially without environmental interactions, and the epigenetic axis involves *learning* through environmental interactions that take place after formation of the individual. As an example, consider the following three paradigms, whose hardware implementations can be positioned along the POE axes: (P) evolutionary algorithms are the (simplified) artificial counterpart of phylogeny in nature, (O) multicellular automata are based on the concept of ontogeny, where a single mother cell gives rise, through multiple divisions, to a multicellular organism, and (E) artificial neural networks embody the epigenetic process, where the system's synaptic weights and perhaps topological structure change through interactions with the environment. Within the domains collectively referred to as soft computing [4], often involving the solution of ill-defined problems coupled with the need for continual adaptation or evolution, the above paradigms yield impressive results, frequently rivaling those of traditional methods.

This paper is an exposition and examination of bio-inspired hardware systems within the POE framework; our goals are: 1) to present an overview of current-day research, 2) to demonstrate that the POE model can be used to classify bio-inspired systems, and 3) to identify possible directions for future research, derived from a POE outlook. We begin in the next section with an examination of the phylogenetic axis (due to space restrictions, and in following the main theme of this TRANSACTIONS, this axis shall be given particular attention). In Section III we present the ontogenetic axis, followed by a discussion of the third axis, epigenesis, in Section IV. Our paper ends in Section V with conclusions and directions for future research, based on the POE model. Specifically, we shall consider the possibilities of combining two axes, along with the ultimate goal of combining all three. This presents a vision for the future which may see the advent of novel hardware systems, inspired by the successful examples provided by nature.

## II. THE PHYLOGENETIC AXIS: EVOLVABLE HARDWARE

In this section we explore the phylogenetic axis of bio-inspired systems, also referred to as evolvable hardware. The main motivation is to attain adaptive systems that are able to accomplish difficult tasks, possibly involving real-time behavior in a complex, dynamical environment. We begin by briefly introducing two underlying themes, artificial evolution and large-scale programmable circuits.

### A. Artificial Evolution

The idea of applying the biological principle of natural evolution to artificial systems, introduced more than three decades ago, has seen impressive growth in the past few years. Usually grouped under the term *evolutionary algorithms* or *evolutionary computation*, we find the domains of genetic algorithms, evolution strategies, evolutionary programming, and genetic programming [5]–[12]. As a generic example of artificial evolution, we consider genetic algorithms [8].

A genetic algorithm is an iterative procedure that consists of a constant-size population of individuals, each one represented by a finite string of symbols, known as the *genome*, encoding a possible solution in a given problem space. This space, referred to as the *search space*, comprises all possible solutions to the problem at hand. The algorithm sets out with an initial population of individuals that is generated at

random or heuristically. At every evolutionary step, known as a *generation*, the individuals in the current population are decoded and evaluated according to some predefined quality criterion, referred to as the *fitness*, or *fitness function*. To form a new population (the next generation), individuals are selected according to their fitness and then transformed via genetically inspired operators, of which the most well known are *crossover* and *mutation*. Iterating this procedure, the genetic algorithm may eventually find an acceptable solution, i.e., one with high fitness.

Evolutionary algorithms are common nowadays, having been successfully applied to numerous problems from different domains, including optimization, automatic programming, machine learning, economics, immune systems, ecology, population genetics, studies of evolution and learning, and social systems [11].

### B. Large-Scale Programmable Circuits

An integrated circuit is called programmable when the user can configure its function by programming. The circuit is delivered after manufacturing in a generic state and the user can adapt it by programming a particular function. In this paper we consider solely programmable *logic* circuits, where the programmable function is a logic one, ranging from simple Boolean functions to complex state machines. The programmed function is coded as a string of bits representing the *configuration* of the circuit. Note that there is a difference between programming a standard microprocessor chip and programming a programmable circuit—the former involves the specification of a sequence of actions, or instructions, while the latter involves a configuration of the machine itself, often at the gate level.

The first programmable circuits allowed the implementation of logic circuits that were expressed as a logic sum of products. These are the PLD's (programmable logic devices), whose most popular version is the PAL (programmable array logic). More recently a novel technology has emerged, affording higher flexibility and more complex functionality: the field-programmable gate array (FPGA) [13]. An FPGA is an array of logic cells placed in an infrastructure of interconnections, which can be programmed at three distinct levels (Fig. 2): 1) the function of the logic cells, 2) the interconnections between cells, and 3) the inputs and outputs. All three levels are configured via a string of bits that is loaded from an external source, either once or several times. In the latter case the FPGA is considered reconfigurable.

FPGA's are highly versatile devices that offer the designer a wide range of design choices. This potential power, however, necessitates a suite of tools to design a system. Essentially, these generate the configuration bit string, given such inputs as a logic diagram or a high-level functional description.

### C. Evolvable Hardware: The Present

If one carefully examines the work carried out to date under the heading evolvable hardware, it becomes evident that this

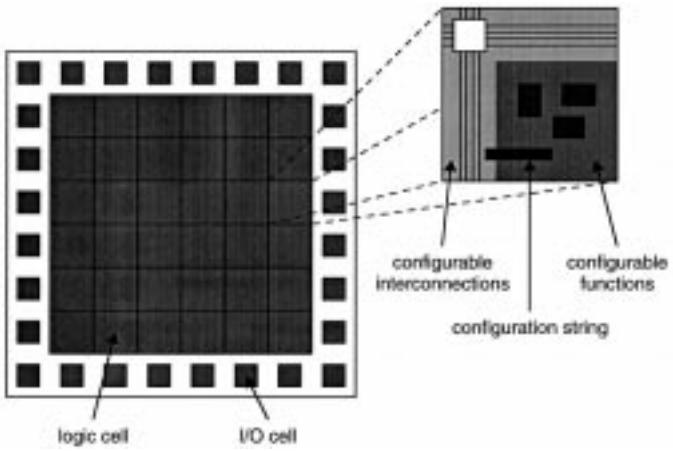


Fig. 2. A schematic diagram of an FPGA. An FPGA is an array of logic cells placed in an infrastructure of interconnections, which can be programmed at three distinct levels: 1) the function of the logic cells, 2) the interconnections between cells, and 3) the inputs and outputs. All three levels are configured via a configuration bit string that is loaded from an external source, either once or several times.

mostly involves the application of evolutionary algorithms to the synthesis of digital systems [14] (recently, Koza *et al.* [15] studied analog systems as well). From this perspective, evolvable hardware is simply a subdomain of artificial evolution, where the final goal is the synthesis of an electronic circuit. The work of Koza [9], which includes the application of genetic programming to the evolution of a three-variable multiplexer and a two-bit adder, may be considered an early precursor along this line. It should be noted that at the time the main goal was that of demonstrating the capabilities of the genetic programming methodology, rather than designing actual circuits. We argue that the term evolutionary circuit design would be more descriptive of such work than that of evolvable hardware (see also [16]). For now, we shall remain with the latter (popular) term; however, we shall return to the issue of clarifying definitions in Section II-E.

Evolvable hardware, taken as a design methodology, offers a major advantage over classical methods. The designer's job is reduced to constructing the evolutionary setup, which involves specifying the circuit requirements, the basic elements, and the testing scheme used to assign fitness (this latter phase is often the most difficult). If these have been well designed, evolution may then (automatically) generate the desired circuit. Currently, most evolved digital designs are suboptimal with respect to traditional methodologies; however, improved results are regularly demonstrated. When examining work carried out to date, one can derive a rough classification of current evolvable hardware, in accordance with the genome encoding (i.e., the circuit description), and the calculation of a circuit's fitness.

*1) Genome Encoding:*

- *High-level languages*. Using a high-level functional language to encode the circuits in question means that the final solution must be transformed to obtain an actual circuit. Thus, such a representation is far removed from the structural (schematic) description. In [9], the evolved

solution is a program describing the (desired) multiplexer or adder rather than an interconnection diagram of logic elements (the actual hardware representation). Hemmi *et al.* [17] used a high-level hardware description language to represent the genomes. Koza *et al.* [15] and Kitano [18] used the rewriting operation, in addition to crossover and mutation, to enable the formation of a hierarchical structure. This is still done within the framework of a high-level language.

- *Low-level languages*. The idea of directly incorporating the bit string representing the configuration of a programmable circuit within the genome was expressed early on by Atmar [19] and more recently by de Garis [20] and Higuchi *et al.* [21]. As a first step one must choose the basic logic gates (e.g., AND, OR, and NOT) and suitably codify them, along with the interconnections between gates, to produce the genome encoding. An example of this approach is offered in [22]. Higuchi *et al.* [23] used a low-level bit string representation of the system's logic diagram to describe small-scale PAL's, where the circuit is restricted to a logic sum of products. The limitations of the PAL circuits have been overcome to a large extent by the introduction of FPGA's, as used, e.g., by Thompson [24].

  The use of a low-level circuit description that requires no further transformation is an important step forward since this potentially enables placing the genome directly in the actual circuit, thus paving the way toward truly evolvable hardware (we shall elaborate upon this point in Section II-E). Until recently, however, FPGA's had presented two major problems: 1) the genome's length was on the order of tens of thousands of bits, rendering evolution practically impossible using current technology, and 2) within the circuit space, consisting of all representable circuits, a large number were invalid (e.g., containing short circuits).

  With the introduction of the new family of FPGA's, the Xilinx 6200 [25], these problems have been reduced. As with previous FPGA families, there is a direct correspondence between the bit string of a cell and the actual logic circuit; however, this now always leads to a viable system (i.e., with no short circuits). Moreover, as opposed to previous FPGA's where one had to configure the entire system, the new family permits the separate configuration of each cell, a markedly faster and more flexible process. Thompson [24] has employed this latter characteristic to reduce the genome's size, without, however, introducing real-time, partial system reconfigurations.

*2) Fitness Calculation:*

- *Off-line evolvable hardware*. The use of a high-level language for the genome representation means that one has to transform the encoded system to evaluate its fitness. This is carried out by simulation, with only the final solution found by evolution actually implemented in hardware. This form of simulated evolution is known as *off-line* evolvable hardware [14].
- *On-line evolvable hardware*. As noted above, the low-level genome representation enables a direct configuration (and reconfiguration) of the circuit, thus entailing the possibility of using real hardware during the evolutionary process. This has been called *on-line* evolution by some of the works found in [14].

### D. Common Features of Current Phylogenetic Hardware

Examining work carried out to date we find many common characteristics that span most current systems, both on-line and off-line, often differing from biological evolution (this difference is not necessarily disparaging, as discussed in Section V).

- Evolution pursues a predefined goal: the design of an electronic circuit, subject to precise specifications. Upon finding the desired circuit, the evolutionary process terminates.
- The population has no material existence. At best, in what has been called on-line evolution, there is one circuit available, onto which individuals from the (off-line) population are loaded *one at a time*, to evaluate their fitness.
- The absence of a real population in which individuals coexist simultaneously entails notable difficulties in the realization of interactions between "organisms." This usually results in a completely independent fitness calculation, contrary to nature which exhibits a coevolutionary scenario.
- If one attempts to resolve a well-defined problem, involving the search for a specific combinatorial or sequential logic system, there are no intermediate approximations. Fitness calculation is carried out by consulting a lookup table which is a complete description of the circuit in question, that must be stored somewhere. This casts some doubts as to the utility of applying an evolutionary process, since one can directly implement the lookup table in a memory device, a solution which may often be faster and cheaper.
- The evolutionary mechanisms are executed outside the resulting circuit. This includes the operators (selection, crossover, mutation) as well as fitness calculation. As for the latter, while what has been advanced as on-line evolution uses a real circuit for fitness evaluation, the fitness values themselves are stored elsewhere.
- The different phases of evolution are carried out sequentially, controlled by a central software unit.

### E. Categorizing Current and Future Evolvable Hardware

The phylogenetic axis admits four qualitative subdivisions (Fig. 3).

- At the bottom of this axis, we find what is in essence *evolutionary circuit design*, where all operations are carried out in software, with the resulting solution possibly loaded onto a real circuit. Though a potentially useful design methodology, this falls completely within the realm of

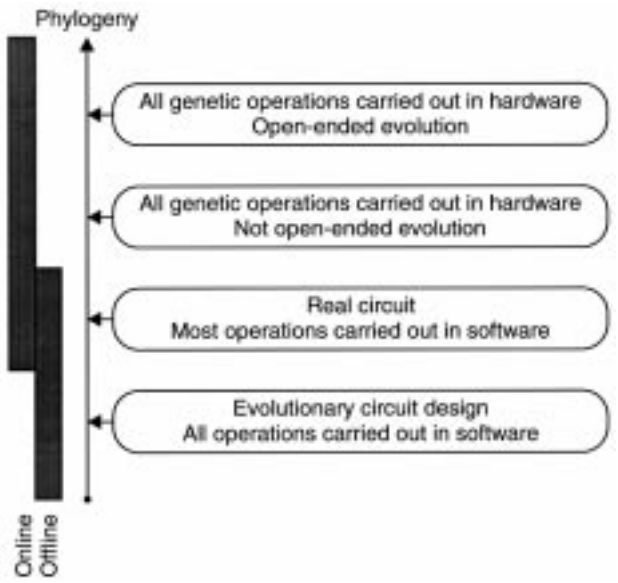


Fig. 3. The phylogenetic axis admits four subdivisions, based on two distinguishing characteristics. The first involves the distinction between *off-line* operations carried out in software, and *on-line* ones which take place on an actual circuit. The second characteristic concerns *open-endedness*. When the fitness criterion is imposed by the user in accordance with the task to be solved, one attains a form of *guided* evolution. This is to be contrasted with *open-ended* evolution occurring in nature, which admits no externally imposed fitness criterion, but rather an implicit, emergent, dynamical one (that could arguably be summed up as survivability).

traditional evolutionary techniques. As examples one can cite the works of [15], [17], [18], and [23].

- Moving upward along the axis, one finds research in which a real circuit is used during the evolutionary process, though most operations are still carried out off-line, in software. Examples are [22] and [26], where fitness calculation is carried out on a real circuit. Thompson *et al.* [22] evolved a hardware controller for a two-wheeled autonomous mobile robot that was required to display simple wall-avoidance behavior in an empty rectangular arena. A standard genetic algorithm was used, with a population of 30 individuals, each one consisting of a 60-bit representation of a dynamic state machine. Thompson [26] evolved an FPGA circuit, consisting of $10 \times 10$ cells, to discriminate between square waves of 1 kHz and 10 kHz presented as inputs. Again, a standard genetic algorithm was employed, with a population of 50 individuals, each one a string of 1800 bits (18 configuration bits per cell), representing a possible circuit. In both cases, a single real circuit was available (one robot in the first experiment and a single FPGA board in the second), with a sequential evaluation of every individual taking place on that circuit, at each evolutionary generation. An interesting aspect of these works concerns the unconstrained use of hardware. While conventional (human) design requires constraints to be applied to the circuit's spatial structure and dynamical behavior, evolution can do away with these. The circuits evolved by [22] and [26] had no spatial structure enforced (e.g., limitations upon recurrent connections), no impositions upon modularity, nor any dynamical constraints such as a synchronizing clock or handshaking between modules.

  Another example that can be situated within this subdivision of the phylogenetic axis is the works of Murakawa *et al.* [27] and Iwata *et al.* [28]. One of the major obstacles which they wished to overcome is that of large genome size (defining the FPGA's configuration). Toward this end they proposed two solutions: 1) variable-length chromosome genetic algorithm (VGA), where the genome does not directly represent the configuration bit string but rather codifies the possible logical operations and interconnections [28]. A decoder is therefore necessary to translate the genome into an FPGA configuration string. This decoder is, however, much simpler than the compilation tools associated with high-level hardware description languages (such as VHDL[1]); therefore this solution reduces the genome's size without incurring a high computational cost. 2) Evolution at the function level, where the basic units are not elementary logic gates (e.g., AND, OR, and NOT), but rather higher-level functions (e.g., sine-wave generator, multiplier) [27]. Since no such commercial FPGA currently exists, they proposed a novel architecture, dubbed $\mathrm{F^2PGA}$ (function-based FPGA). One can combine both solutions, using VGA encoding with an $\mathrm{F^2PGA}$ architecture.

  The methodologies proposed by [27] and [28] were targeted at applications involving real-time adaptation in a changing environment. This raises a problem regarding fitness evaluation: since each individual in the (off-line) population is sequentially downloaded onto the (single) real circuit available, real-time behavior cannot be attained during the evolutionary phase, but only upon its termination. Their solution to this problem was to use *two* real circuits, one submerged within the environment in question, executing the configuration of the best evolved individual so far, the other serving for (sequential) fitness calculation. This two-circuit system enables the use of an actual circuit, while additionally exhibiting real-time behavior.

  It is important to note that while experiments of the above type have been referred by some as on-line evolution, there is a prominent off-line aspect since the population is stored in an external computer, which also controls the evolutionary process. It would probably be more appropriate to reserve the term on-line for the next subdivision.
- Still further along the phylogenetic axis, one finds systems in which all operations (selection, crossover, mutation), as well as fitness evaluation, are carried out on-line, in hardware. The major aspect missing concerns the fact that evolution is not open ended, i.e., there is a predefined goal and no dynamic environment to speak of. An example is the work of Goeke *et al.* [30], where an evolving cellular system was implemented in which evolution takes place

[1] Very High Speed Integrated Circuit (VHSIC) Hardware Description Language. See, e.g., [29].

completely on-line. It is based on the cellular automata model, a discrete dynamical system that performs computations in a distributed fashion on a spatially extended grid. A cellular automaton consists of an array of cells, each of which can be in one of a finite number of possible states, updated synchronously in discrete time steps according to a *local, identical* interaction rule [31], [32]. The *state* of a cell at the next time step is determined by the current states of a surrounding neighborhood of cells. This transition is usually specified in the form of a *rule table*, delineating the cell's next state for each possible neighborhood configuration. The cellular array (grid) is $n$-dimensional, where $n = 1, 2, 3$ is used in practice. Nonuniform cellular automata have also been considered in which the local update rule need not be identical for all grid cells [33], [34].

Based on the *cellular programming* evolutionary algorithm of Sipper [34] (see also [35]–[39]), Goeke *et al.* [30] implemented an evolving, one-dimensional, nonuniform cellular automaton. Each of the system's 56 binary-state cells contains a genome that represents its rule table. These genomes are initialized at random, thereupon to be subjected to evolution. The environment imposed on the system specifies the resolution of a global synchronization task: upon presentation of a random initial configuration of cellular states, the system must reach, after a bounded number of time steps, a configuration whereupon the states of the cells oscillate between all zeros and all ones on successive time steps. This may be compared to a swarm of fireflies, thousands of which may flash on and off in unison, having started from totally uncoordinated flickerings. Each insect has its own rhythm, which changes only through local interactions with its neighbors' lights [40] (for a review on other phenomena in nature involving synchronous oscillation, see [41]). Due to the local connectivity of the system, this global behavior involving the entire grid comprises a difficult task. Nonetheless, applying the evolutionary process of [34], the system evolves (i.e., the genomes change) such that the task is solved [30].

The evolving cellular system described in the previous paragraph exhibits complete on-line evolution, all operators being carried out in hardware with no reference to an external computer. This demonstrates that evolving ware, *evolware* [30], [37], can be constructed (Fig. 4). The evolware board was implemented using FPGA circuits, configured such that each cell within the system behaves in a certain general manner, after which evolution is used to find the cell's specific behavior, i.e., its rule table. Thus, the system consists of a fixed part and an evolving part, both specified via FPGA configuration strings. An interesting outlook on this setup is to consider the evolutionary process as one where an organism evolves within a given species, the former specified by the FPGA's evolving part, the latter specified by the fixed part. This raises the interesting issue of evolving the species itself.

- The last subdivision, situated at the top of the phylogenetic axis, involves a population of hardware entities evolving in an open-ended environment. When the fitness criterion is imposed by the user in accordance with the task to be solved (currently the rule with artificial evolution techniques), one attains a form of guided, or directed, evolution. This is to be contrasted with open-ended evolution occurring in nature, which admits no externally imposed fitness criterion, but rather an implicit, emergent, dynamical one (that could arguably be summed up as survivability). Open-ended, undirected evolution is the only form of evolution known to produce such devices as eyes, wings, and nervous systems and to give rise to the formation of species. Undirectedness may have to be applied to artificial evolution if we want to observe the emergence of completely novel systems.

We argue that only the last category can be truly considered evolvable hardware, a goal which still eludes us at present. We point out that a more correct term would probably be evolving hardware. A natural application area for such systems is within the field of autonomous robots, which involves machines capable of operating in unknown environments without human intervention [42]. A related application domain is that of controllers for noisy, changing environments. Another interesting example would be what we call Hard-Tierra, involving the hardware implementation (e.g., using FPGA circuits) of the Tierra "world," which consists of an open-ended environment of evolving computer programs [43].[2] A small-scale experiment along this line was undertaken in [44]. The idea of Hard-Tierra is important since it demonstrates that open-endedness does not necessarily imply a real, biological environment.

## III. The Ontogenetic Axis: Replicating and Regenerating Hardware

The ontogenetic axis involves the *development* of a single individual from its own genetic material, essentially without environmental interactions. As can be seen in Fig. 5 (based on [45] and [46]) ontogeny can be considered orthogonal to phylogeny. The main process involved in the ontogenetic axis can be summed up as *growth*, or *construction*. Ontogenetic hardware exhibits such characteristics as replication and regeneration which find their use in many applications. For example, replicating systems have the ability to self-repair upon suffering heavy damage [47] and have been proposed as an economical means of space exploration [48]. Replication can in fact be considered a special case of growth—this process involves the creation of an identical organism by

[2] *Tierra* is a virtual world, consisting of computer programs that can undergo evolution. In contrast to evolutionary algorithms where fitness is defined by the user, the Tierra "creatures" (programs) receive no such direction. Rather, they compete for the natural resources of their computerized environment, namely, CPU time and memory. Since only a finite amount of these are available, the virtual world's natural resources are limited, as in nature, giving rise to competition between creatures. Ray observed the formation of an "ecosystem" within the Tierra world, including organisms of various sizes, parasites, and hyper-parasites [43].


Fig. 4. The firefly evolware board. The system is an evolving, one-dimensional, nonuniform cellular automaton. Each of the 56 cells contains a genome that represents its rule table; these genomes are randomly initialized, thereupon to be subjected to evolution. The board contains the following components: 1) LED indicators of cell states (top), 2) switches for manually setting the initial states of cells (top, below LED's), 3) Xilinx FPGA chips (below switches), 4) display and knobs for controlling two parameters ("time steps" and "configurations") of the cellular programming algorithm (bottom left), 5) a synchronization indicator (middle left), 6) a clock pulse generator with a manually adjustable frequency from 0.1 Hz to 1 MHz (bottom middle), 7) an LCD display of evolved rule tables and fitness values obtained during evolution (bottom right), and 8) a power-supply cable (extreme left). (Note that this is the system's sole external connection.)

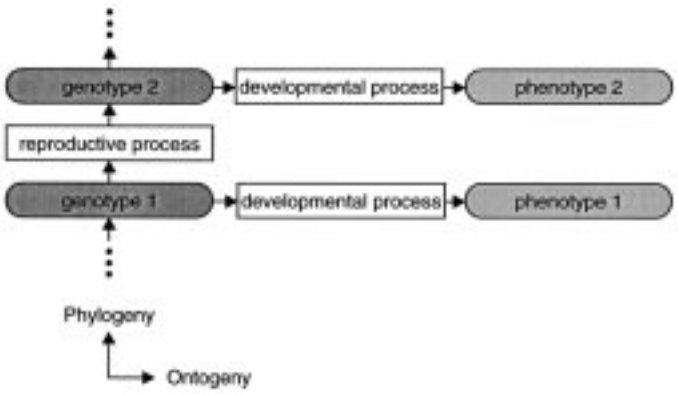


Fig. 5. The phylogenetic and ontogenetic axes can be considered orthogonal. The figure shows two generations preceded and followed by an indefinite number of generations. Ontogeny involves the development of the phenotype in a given generation (horizontal arrows), while phylogeny involves the succession of generations through reproduction of the genotype (vertical arrows). Note that genes, the basic constituents of the genome, act on two quite different levels: they participate in the developmental process, influencing the development of the phenotype in a given generation, and they participate in genetics, having themselves copied down the generations (reproduction) [46].

duplicating the genetic material of a mother entity onto a daughter one, thereby creating an exact clone. It is important to distinguish between two distinct terms, replication and reproduction, which are often considered synonymous. Replication is an ontogenetic, developmental process, involving no genetic operators, resulting in an exact duplicate of the parent organism (as in the budding process of the hydra, described by [49]). Reproduction, on the other hand, is a phylogenetic process, involving genetic operators such as crossover and mutation, thereby giving rise to variety and ultimately to evolution (note that reproduction has been justly placed on the vertical axis of Fig. 5).

Research on ontogenetic systems began with von Neumann's work in the late 1940's on self-replicating machines. This work was later extended by others, and more recently we have seen the emergence of systems that exhibit other ontogenetic mechanisms, such as cellular division and cellular differentiation. This line of research can be divided into four stages, placed along the ontogenetic axis (Fig. 6).

- Von Neumann [50] (see also [51]) and his successors Banks [52], Burks [53], and Codd [54] developed self-replicating automata capable of universal computation (i.e., able to simulate a universal Turing machine [55]) and of universal construction (i.e., able to construct any automaton described by an artificial genome). While

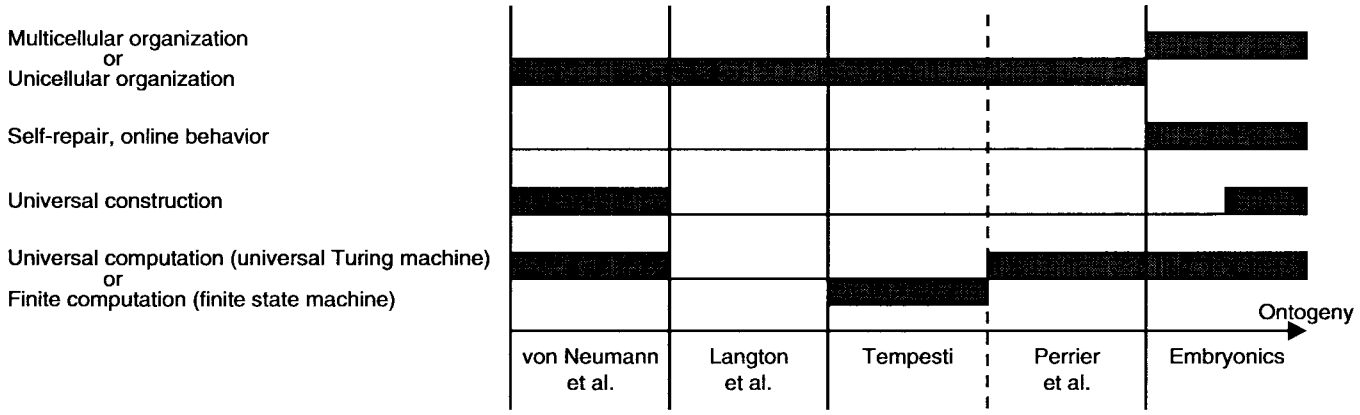


Fig. 6. The ontogenetic axis admits four stages, based on a number of distinguishing characteristics: universal computation (the ability to simulate a universal Turing machine), universal construction (the ability to construct any automaton described by an artificial genome), self-repair capabilities, and unicellular or multicellular organization.

the complexity of these automata is such that no full physical implementation has yet been possible, the von Neumann cell has recently been implemented in hardware [56].

- Langton [57] and his successors Byl [58], Reggia *et al.* [59], and Morita and Imai [60] developed self-replicating automata which are much simpler and which have been simulated in their entirety. These machines, however, lack any computing and constructing capabilities, with their sole functionality being that of self-replication.
- Tempesti [61] and Perrier *et al.* [62] developed self-replicating automata inspired by Langton's work, yet endowed with finite [61] or universal [62] computational capabilities.
- One of the defining characteristics of a biological cell concerns its role as the smallest part of a living being which carries the complete plan of the being, that is its genome [63]. In this respect, the above self-replicating automata are unicellular organisms: there is a single genome describing (and contained within) the entire machine.

  Mange *et al.* [47], [63], [64] and Marchal *et al.* [65] proposed a new architecture called *embryonics*, or embryonic electronics. Based on three features usually associated with the ontogenetic process in living organisms, namely, multicellular organization, cellular differentiation, and cellular division, they introduced a new cellular automaton, complex enough for universal computation, yet simple enough for a physical implementation through the use of commercially available digital circuits. They developed an artificial cell, a *biodule* (biological module), comprising three structures found in living cells (Fig. 7) [49]: 1) a plastic box constitutes the external *membrane*, ensuring the cell's material encasement and realizing all the electronic functions necessary for communication with neighboring cells, 2) a processor responsible for interpreting the genome constitutes the *cytoplasm*, in analogy to a ribosome, and 3) a random access memory (RAM) acts as the cell's *nucleus*, containing a copy of the entire genetic makeup, i.e., a genome composed of a linear sequence of genes.

  The biodule cell is used as an elementary unit from which multicellular organisms can ontogenetically develop to perform useful tasks (e.g., act as universal Turing machines [66], [67]). Cellular differentiation takes place by having each cell compute its coordinates (i.e., position) within a two-dimensional space, after which it can extract the specific gene within the genome responsible for the cell's functionality. Cellular division occurs when a mother cell, the *zygote*, arbitrarily placed within the two-dimensional grid, multiplies to fill a large portion of the space, thus forming a multicellular organism. In addition to self-replication, this artificial organism also exhibits self-repair capabilities, another biologically inspired phenomenon. To embed universal construction, Mange *et al.* are designing the basic cell with a molecular organization, similar to that of the transcription-translation mechanism (ribosome) [67]. Such self-replicating machines are *multicellular* artificial organisms, in the sense that each of the several cells comprising the organism contains one copy of the complete genome.

Several other works can be placed along the ontogenetic axis, including, e.g., L-systems [68], cellular encoding [69], graph generation systems [70], and self-replicating programs [71]. We have not discussed these at length as they are currently implemented solely in software, while our emphasis is on hardware systems.

## IV. The Epigenetic Axis: Learning Hardware

The epigenetic axis involves learning through environmental interactions that take place after formation of the individual. To the best of our knowledge, there exist three major epigenetic systems in living multicellular organisms: the nervous system, the immune system, and the endocrine system, with the first two having already served as inspiration for engineers. The nervous system has received the most attention, giving rise to the field of artificial neural networks. This will be

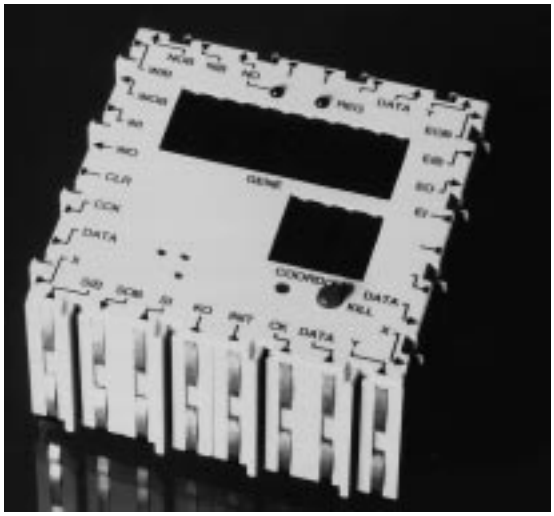

Fig. 7. An artificial cell: the biodule. A processor responsible for interpreting the genome constitutes the cytoplasm, in analogy to a ribosome, along with a RAM acting as the cell's nucleus, containing a copy of the entire genetic makeup, i.e., the genome. Displayed on the top cover are the cell's coordinates, as well as the specific gene within the genome that determines its functionality; these are acquired during cellular differentiation. The KILL button is used to induce the self-repair (regeneration) mechanism.

the focus of our discussion below. The immune system has inspired systems for detecting software errors [72], controllers for mobile robots [73], and immune systems for computers [74]. Immunity of living organisms is a major domain of biology. It has been demonstrated that the immune system is capable of learning, recognizing, and, above all, eliminating foreign bodies which continuously invade the organism. Moreover, when viewed from the engineering standpoint, it is most interesting that immunity is maintained when faced with a dynamically changing environment. This feature leads us to surmise that the immune system, if implemented as an engineering model, can provide a new tool suitable for confronting dynamic problems, involving unknown, possibly hostile, environments. The human endocrine system is made up of a large number of glandular tissues that have in common the fact that they secrete directly into the blood stream chemical messengers, known as hormones, that regulate and integrate bodily functions (such as reproduction). This system resembles in some of its functionalities the nervous system in that both help the individual cope with changes in its environment.

The nervous system remains the most popular epigenetic source of inspiration for engineers. From a biological point of view, it has been determined that the genome contains the formation rules that specify the outline of the nervous system [1], [2]. It is primarily the synapses, the zones of contact between two neurons, where learning takes place, through interactions with the environment during the organism's lifetime. The nervous system of living organisms thus represents a mixture of the innate and the acquired, the latter precluding the possibility of its direct hereditary transmission under Darwinian evolution.

A predominant approach in the field of artificial neural networks consists of applying a learning algorithm to the

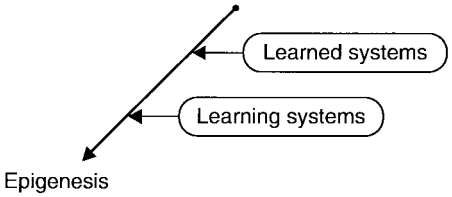


Fig. 8. The epigenetic axis: moving from learned (instinctive) systems to on-line learning networks.

modification of synaptic weights, using a predesigned network topology. A prime difference between simple rote learning and intelligent learning is the generalization process taking place in the latter. One can view a predesigned network as an implementation of a learned system that exhibits instinctive behavior [75]. Indeed, there is growing evidence that the human brain has many more such instinctive networks than is usually acknowledged [76], [77], possibly due to their being faster and less resource-demanding with respect to learning systems, which adapt continuously within a dynamic environment. Learning networks exhibit the plasticity necessary to confront complex, dynamical tasks and must be able to adapt at two distinct levels, changing the dynamics of interneuron interactions (usually through changes in synaptic weights) as well as modifying the network topology itself. Topology modification has proven to be a successful solution to a problem known as the stability–plasticity dilemma, i.e., how can a learning system preserve what it has previously learned, while continuing to incorporate new knowledge [78]. Evolution may ultimately replace such learning networks by instinctive ones, e.g., via the Baldwin effect, whereby a learned (acquired) behavior becomes embedded within the organism's genome (i.e., its innate behavioral repertoire) through evolution [79]–[81] (this may be considered a melange of phylogeny and epigenesis, an issue which shall be expanded upon in Section V).

Artificial neural networks have been implemented many times, mostly in software rather than in hardware, though only the latter concern us here. On-line learning is essential if one wishes to obtain learning systems as opposed to merely learned ones (Fig. 8). While neural network hardware had already appeared in the 1980's [82], [83], only today are we seeing the birth of the technology that enables true on-line learning.

Several network models with modifiable topologies have been proposed, including growing neural networks [84], neural networks with incremental learning algorithms [85], and construction algorithms [86]. To our knowledge, none of these have been implemented as on-line hardware, probably due to the high cost incurred. Recently, a number of FPGA-based neural network systems have appeared, though without dynamic restructuring [87]–[89]. Perez and Sanchez [90], [91] have developed a network architecture dubbed *FAST* (flexible adaptable-size topology) that implements an unsupervised clustering algorithm, where the network must discover correlations within the input data and cluster, or categorize them accordingly. The network's topology changes dynamically,

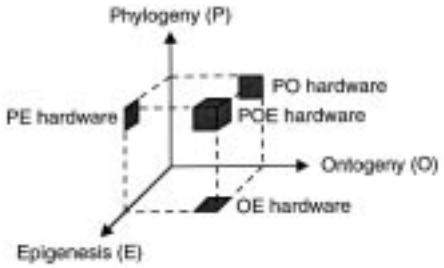


Fig. 9. Combining POE axes to create novel bio-inspired systems: The PO plane involves evolving hardware that exhibits ontogenetic characteristics, such as growth, replication, and regeneration, the PE plane includes, e.g., evolutionary artificial neural networks, the OE plane combines ontogenetic mechanisms (self-replication, self-repair) with epigenetic (e.g., neural network) learning, and finally, the POE space comprises systems that exhibit characteristics pertaining to all three axes. An example of the latter would be an artificial neural network (epigenetic axis), implemented on a self-replicating multicellular automaton (ontogenetic axis), whose genome is subject to evolution (phylogenetic axis).

with a new neuron being added when a sufficiently distinct input pattern is encountered, and an active neuron being deleted through the application of probabilistic deactivation. Each neuron maintains an $n$-dimensional reference vector and a threshold, both of which determine its sensitivity region, i.e., the input vectors to which it is maximally sensitive. The network adapts to a particular environment (problem) through an interplay between three processes: 1) classical learning (through modifications to reference vectors and thresholds), 2) incremental growth (addition of new neurons), and 3) pruning (deactivation of active neurons). To date, a four-neuron prototype has been implemented using FPGA's and FPIC's (field-programmable interconnection circuits) and applied to the solution of pattern recognition and enhancement problems [90], [91]. This latter approach is currently being applied to the construction of an on-line robotic neurocontroller. Moreno [92] examined a number of topology-restructuring neural network algorithms with respect to their amenability to VLSI implementation. Other interesting paths are those that combine two or three axes of the POE model, as discussed in the next section.

## V. Conclusions: Toward Novel Bio-Inspired Hardware Systems

We presented the POE model for classifying bio-inspired hardware systems, based on three axes: phylogeny, ontogeny, and epigenesis (Fig. 1). We described each axis and provided examples of systems situated along them.[3] A natural extension which suggests itself is the combination of two, and ultimately all three axes, to attain novel bio-inspired hardware (Fig. 9). As examples we propose the following.

- *The PO plane*. This involves evolving hardware that exhibits ontogenetic characteristics, such as growth, replication, and regeneration. For example, Sipper and Tomassini [36], [93] evolved nonuniform cellular automata to act as random number generators. Mange *et al.* [47] showed that such evolved generators can be implemented by a multicellular automaton that exhibits self-replication and self-repair. Thus, the eventual combination of these two projects can be considered to be in the phylogenetic-ontogenetic plane.
- *The PE plane*. The architecture of the brain is the result of a long evolutionary process, during which a large set of specialized subsystems interactively emerged, carrying out tasks necessary for survival and reproduction [94]. Learning (epigenesis) in biological neural systems can be considered to serve as a mechanism for fine-tuning these broadly laid out neural circuits [95]. Although it is impossible that the genes code all structural information about the brain (Section I), they may be the ultimate determinant of what it can and cannot learn [96].

  The idea of evolutionary, artificial neural networks, situated in the PE plane, has received attention in recent years. This involves a population of neural networks, where evolution takes place at the global (population) level, with learning taking place at the individual (neural network) level. Examples are the works of Liu and Yao [97], Nolfi *et al.* [98], and Yao [99], though they are currently completely off-line. Another interesting (natural) example that may be considered to reside within the PE plane is that of the Baldwin effect, which exhibits an intricate interplay between phylogeny and epigenesis (see Section IV). The use of this process in simulated systems has been explored, e.g., by [81] and [100]. As a final case in point regarding the PE plane in nature one can consider the issue of language acquisition in human beings, specifically, to what extent is this remarkable ability innate (phylogenetic) or acquired (epigenetic). (For a recent discussion and brief historical perspective, see Dennett [101, ch. 13]. For an exploration of this issue in artificial settings, see [102]–[104].)
- *The OE plane*. According to selectionism (e.g., [105]), selective pressures operate on epigenetic variation during the ontogeny of the individual (in "somatic" time), not on a phylogenetic time scale [106]. This suggests the possibility of combining the ontogenetic mechanisms discussed above, with the epigenetic (neural network) learning algorithms. As an example, one can cite the works of de Garis [107] and Gers and de Garis [108], which involve the ontogenetic growth of a neural network that can then undergo epigenetic learning. Currently, only the ontogenetic part has been reported upon, with learning yet to be demonstrated. The FAST neural network with its dynamically changing topology (Section IV) can be extended into the OE plane by obtaining the topology via an ontogenetic process (e.g., as proposed by [70]).

  Inductive learning can be interpreted as the capability to infer a response to an unknown situation, achieved through generalizing from previously encountered, known situations. Engineers are perpetually confronted with a tradeoff between generalization and robustness. While adding a multitude of neurons increases the system's

[3] The POE model was recently used to classify all the works presented at a conference on evolvable systems [3].

fault tolerance, there is a risk of overfitting if we do not attempt to generalize [109]. Implementing neurons in hardware is generally quite expensive, so it is imperative that cost-effectiveness be considered, trying to obtain the smallest possible network. Once a good generalization is obtained (with respect to a certain problem or situation), fault tolerance can be achieved through other self-repair mechanisms, e.g., those used by the embryonics system (Section III).

- *The POE space*. The development of an artificial neural network (epigenetic axis), implemented on a self-replicating multicellular automaton (ontogenetic axis), whose genome is subject to evolution (phylogenetic axis), constitutes a possible example situated in the POE space (Fig. 9).

Bio-inspired hardware systems to date mostly exhibit a clear separation between the POE axes, each plainly situated along a single one. Possible extensions were noted above, including systems that reside in one of the three planes, and eventually within the POE space. In fact, the success of the "POE community's" endeavor can be measured by the disappearance (extinction?) of the POE model (or rather its utility). This will ensue from the composition of future systems which may eventually exhibit characteristics of all three axes, i.e., reside within the POE space. At such time, the POE model will have outlived its usefulness.

Note that the systems considered in this paper are bio-inspired. This means that, while motivated by observations of nature, strict adherence to her solutions is not a *sine qua non*. As an example, consider the issue of Lamarckian evolution, which involves the direct inheritance of acquired characteristics. While the biological theory of evolution has shifted from Lamarckism to Darwinism, this does not preclude the use of artificial Lamarckian evolution [110]. Another example concerns the time scales of natural processes, where phylogenetic changes occur at much slower rates than either ontogenetic or epigenetic ones, a characteristic which need not necessarily hold in our case. Thus, deviations from what is strictly natural may definitely be of use in bio-inspired systems.

Comparing our current artificial systems with those found in nature is beyond the scope of this paper. Moreover, this would require devising a sophisticated scheme, involving a set of comparison criteria and measures. Nonetheless, it is interesting to consider one aspect, that of genome size. Table I presents the genome sizes of some natural and artificial systems, clearly demonstrating our present, comparatively primitive, state. Even one of the most complex artificial systems (in terms of genome size), namely, von Neumann's self-replicating universal constructor, is almost two orders of magnitude smaller than a simple bacterium. Furthermore, this system was not evolved, but rather designed by hand, and is highly intolerant of errors—inversion of a single bit in the genome will almost always result in the machine's complete destruction. For the evolved systems it is evident that we are still far removed from even the simplest natural organisms.

TABLE I
GENOME SIZE AND DNA CONTENT (OR ITS ARTIFICIAL ANALOG). (THE DATA CONCERNING THE NATURAL SYSTEMS IS BASED ON [113] AS QUOTED IN [114]. THE ESTIMATE OF THE SIZE OF VON NEUMANN'S SELF-REPLICATING UNIVERSAL CONSTRUCTOR IS BASED ON [51])

| "Organism" | Genome size (bits) | Coding DNA (%) |
|---|---|---|
| Firefly (Sec. II-E) | $4.48 \times 10^2$ | 100 |
| Frequency discriminator (Sec. II-E) | $1.8 \times 10^3$ | 100 |
| Embryonics Turing machine (Sec. III) | $2.432 \times 10^3$ | 100 |
| Von Neumann's universal constructor | $1.0 \times 10^5$ | 100 |
| Bacterium (*Escherichia coli*) | $8.0 \times 10^6$ | 100 |
| Nematode (*Caenorhabditis*) | $1.8 \times 10^8$ | 25 |
| Fruit fly (*Drosophila*) | $3.6 \times 10^8$ | 33 |
| Human | $7.0 \times 10^9$ | 9-27 |
| Flowering plant (*Fritillaria*) | $2.6 \times 10^{11}$ | 0.02 |

Note that not only is genome size much smaller for artificial systems, there is also an absence of noncoding "DNA," a prominent feature of natural organisms. The importance of this phenomenon is not yet fully understood (for studies of this issue within an artificial evolution framework see [111] and [112]).

On a more reflective (and somewhat philosophical) note, one can consider the works described herein as a collective effort in what amounts, perhaps, to the origins of a new biology. The evolutionary history of our planet has seen the nascence and, more prominently, extinction, of untold numbers of species. An analogous process is taking place at this very instant in several research facilities around the globe. Artificial species, such as those described in this paper, are being born (currently through design, but ultimately, perhaps, via evolution), to be immediately subjected to competition with their congeners, giving rise to selection forces at the species level. In analogy to nature, the years to come may see the appearance and disappearance of entire artificial species. Granted, we are currently at the "primordial-soup" stage, yet it would be interesting to take, especially at this early point, a historical stance. Our current position is unique in that we can trace out the entire phylogenetic tree of this new biology, including the discontinued kingdoms, phyla, classes, orders, families, genera, and species, thereby obtaining a complete history, or lineage. This is in contrast to nature, where we can observe only those species that have survived the millenia, or have left prominent footmarks in the fossil record (see [101] for interesting discussions on some of the issues pertaining to natural evolution).

Looking (and dreaming) toward the future, one can imagine nano-scale (bioware) systems becoming a reality, which will be endowed with evolutionary, reproductive, regenerative, and learning capabilities. Such systems could give rise to novel species which will coexist alongside carbon-based organisms.

## ACKNOWLEDGMENT

The authors are grateful to A. Danchin, D. B. Fogel, J. R. Koza, and the anonymous reviewers for their careful reading of this manuscript and their many helpful remarks

and suggestions. They also thank A. Badertscher for his help in obtaining the firefly image of Fig. 4.

## REFERENCES


[1] A. Danchin, "A selective theory for the epigenetic specification of the monospecific antibody production in single cell lines," *Ann. Immunol. (Institut Pasteur)*, vol. 127C, pp. 787–804, 1976.

[2] ———, "Stabilisation fonctionnelle et épigénèse: une approche biologique de la genèse de l'identité individuelle," in *L'identité*, J.-M. Benoist, Ed. Paris, France: Grasset, 1977, pp. 185–221.

[3] E. Sanchez, D. Mange, M. Sipper, M. Tomassini, A. Pérez-Uribe, and A. Stauffer, "Phylogeny, ontogeny, and epigenesis: Three sources of biological inspiration for softening hardware," in *Proc. 1st Int. Conf. Evolvable Systems: From Biology to Hardware (ICES96)* (Lecture Notes in Computer Science). Heidelberg, Germany: Springer-Verlag, 1997.

[4] R. R. Yager and L. A. Zadeh, *Fuzzy Sets, Neural Networks, and Soft Computing*. New York: Van Nostrand Reinhold, 1994.

[5] T. Bäck, *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*. New York, Oxford Univ. Press, 1996.

[6] D. B. Fogel, *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence*. Piscataway, NJ: IEEE Press, 1995.

[7] D. E. Godlberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Reading, MA: Addison-Wesley, 1989.

[8] J. H. Holland, *Adaptation in Natural and Artificial Systems*. Ann Arbor, MI: Univ. of Michigan Press, 1975.

[9] J. R. Koza, *Genetic Programming*. Cambridge, MA: MIT Press, 1992.

[10] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, 3rd ed. Heidelberg, Germany: Springer-Verlag, 1996.

[11] M. Mitchell, *An Introduction to Genetic Algorithms*. Cambridge, MA: MIT Press, 1996.

[12] H.-P. Schwefel, *Evolution and Optimum Seeking*. New York: Wiley, 1995.

[13] E. Sanchez, "Field-Programmable Gate Array (FPGA) circuits," in *Toward Evolvable Hardware* (Lecture Notes in Computer Science, vol. 1062), E. Sanchez and M. Tomassini, Eds. Heidelberg, Germany: Springer-Verlag, 1996, pp. 1–18.

[14] E. Sanchez and M. Tomassini, Eds., *Toward Evolvable Hardware* (Lecture Notes in Computer Science, vol. 1062). Heidelberg, Germany: Springer-Verlag, 1996,

[15] J. R. Koza, F. H. Bennett III, D. Andre, and M. A. Keane, "Automated WYWIWYG design of both the topology and component values of electrical circuits using genetic programming," in *Genetic Programming 1996: Proceedings of the First Annual Conference*, J. R. Koza, D. E. Goldberg, D. B. Fogel, and R. L. Riolo, Eds. Cambridge, MA: MIT Press, 1996, pp. 123–131.

[16] X. Yao and T. Higuchi, "Promises and challenges of evolvable hardware," in *Proc. 1st Int. Conf. Evolvable Systems: From Biology to Hardware (ICES96)* (Lecture Notes in Computer Science). Heidelberg, Germany: Springer-Verlag, 1997.

[17] H. Hemmi, J. Mizoguchi, and K. Shimohara, "Development and evolution of hardware behaviors," in *Toward Evolvable Hardware* (Lecture Notes in Computer Science, vol. 1062), E. Sanchez and M. Tomassini, Eds. Heidelberg, Germany: Springer-Verlag, 1996, pp. 250–265.

[18] H. Kitano, "Morphogenesis for evolvable systems," in *Toward Evolvable Hardware* (Lecture Notes in Computer Science, vol. 1062), E. Sanchez and M. Tomassini, Eds. Heidelberg, Germany: Springer-Verlag, 1996, pp 99–117.

[19] J. W. Atmar, "Speculation on the evolution of intelligence and its possible realization in machine form," Ph.D. dissertation, New Mexico State University, Las Cruces, 1976.

[20] H. de Garis, "Evolvable hardware: Genetic programming of a Darwin machine," in *Artificial Neural Nets and Genetic Algorithms*, R. F. Albrecht, C. R. Reeves, and N. C. Steele, Eds. Heidelberg: Springer-Verlag, 1993, pp. 441–449.

[21] T. Higuchi, T. Niwa, T. Tanaka, H. Iba, H. de Garis, and T. Furuya, "Evolving hardware with genetic learning: A first step toward building a Darwin machine," in *Proc. 2nd Int. Conf. Simulation of Adaptive Behavior (SAB92)*. Cambridge, MA: MIT Press, 1993, pp. 417–424.

[22] A. Thompson, I. Harvey, and P. Husbands, "Unconstrained evolution and hard consequences," in *Toward Evolvable Hardware* (Lecture Notes in Computer Science, vol. 1062), E. Sanchez and M. Tomassini, Eds. Heidelberg: Springer-Verlag, 1996, pp. 136–165.

[23] T. Higuchi, M. Iwata, I. Kajitani, H. Iba, Y. Hirao, T. Furuya, and B. Manderick, "Evolvable hardware and its application to pattern recognition and fault-tolerant systems," in *Toward Evolvable Hardware*, (Lecture Notes in Computer Science, vol. 1062), E. Sanchez and M. Tomassini, Eds. Heidelberg: Springer-Verlag, 1996, pp. 118–135.

[24] A. Thompson, "Silicon evolution," in *Genetic Programming 1996: Proceedings of the First Annual Conference*, J. R. Koza, D. E. Goldberg, D. B. Fogel, and R. L. Riolo, Eds. Cambridge, MA: MIT Press, 1996, pp. 444–452.

[25] Xilinx, *The Programmable Logic Data Book*, San Jose, CA, 1996.

[26] A. Thompson, "An evolved circuit, intrinsic in silicon, entwined with physics," in *Proc. 1st Int. Conf. Evolvable Systems: From Biology to Hardware (ICES96)* (Lecture Notes in Computer Science). Heidelberg: Springer-Verlag, 1997.

[27] M. Murakawa, S. Yoshizawa, I. Kajitani, T. Furuya, M. Iwata, and T. Higuchi, "Hardware evolution at function level," in *Parallel Problem Solving from Nature—PPSN IV* (Lecture Notes in Computer Science, vol. 1141), H.-M. Voigt, W. Ebeling, I. Rechenberg, and H.-P. Schwefel, Eds. Heidelberg: Springer-Verlag, 1996, pp. 62–71.

[28] M. Iwata, I. Kajitani, H. Yamada, H. Iba, and T. Higuchi, "A pattern recognition system using evolvable hardware," in *Parallel Problem Solving from Nature—PPSN IV* (Lecture Notes in Computer Science, vol. 1141), H.-M. Voigt, W. Ebeling, I. Rechenberg, and H.-P. Schwefel, Eds. Heidelberg: Springer-Verlag, 1996, pp. 761–770.

[29] A. Rushton, *VHDL for Logic Synthesis: An Introductory Guide for Achieving Design Requirements*. London: McGraw-Hill, 1995.

[30] M. Goeke, M. Sipper, D. Mange, A. Stauffer, E. Sanchez, and M. Tomassini, "Online autonomous evolware," in *Proc. 1st Int. Conf. Evolvable Systems: From Biology to Hardware (ICES96)* (Lecture Notes in Computer Science). Heidelberg: Springer-Verlag, 1997.

[31] S. Wolfram, "Statistical mechanics of cellular automata," *Rev. Mod. Phys.*, vol. 55, no. 3, pp. 601–644, July 1983.

[32] T. Toffoli and N. Margolus, *Cellular Automata Machines*. Cambridge, MA: MIT Press, 1987.

[33] M. Sipper, "Non-uniform cellular automata: Evolution in rule space and formation of complex structures," in *Artificial Life IV*, R. A. Brooks and P. Maes, Eds. Cambridge, MA: MIT Press, 1994, pp. 394–399.

[34] ———, *Evolution of Parallel Cellular Machines: The Cellular Programming Approach*. Heidelberg: Springer-Verlag, 1997.

[35] ———, "Co-evolving nonuniform cellular automata to perform computations," *Physica D*, vol. 92, pp. 193–208, 1996.

[36] M. Sipper and M. Tomassini, "Generating parallel random number generators by cellular programming," *Int. J. Mod. Phys. C*, vol. 7, no. 2, pp. 181–190, 1996.

[37] M. Sipper, "Designing evolware by cellular programming," in *Proc. 1st Int. Conf. Evolvable Systems: From Biology to Hardware (ICES96)* (Lecture Notes in Computer Science). Heidelberg: Springer-Verlag, 1997.

[38] M. Sipper and E. Ruppin, "Co-evolving architectures for cellular machines," *Physica D*, vol. 99, pp. 428–441, 1997.

[39] M. Sipper, "Evolving uniform and nonuniform cellular automata networks," in *Annual Reviews of Computational Physics*, vol. V, D. Stauffer, Ed. Singapore: World Scientific, 1997.

[40] J. Buck, "Synchronous rhythmic flashing of fireflies II," *Q. Rev. Biology*, vol. 63, no. 3, pp. 265–289, Sept. 1988.

[41] S. H. Strogatz and I. Stewart, "Coupled oscillators and biological synchronization," *Sci. Amer.*, vol. 269, no. 6, pp. 102–109, Dec. 1993.

[42] R. A. Brooks, "New approaches to robotics," *Science*, vol. 253, no. 5025, pp. 1227–1232, Sept. 1991.

[43] T. S. Ray, "An approach to the synthesis of life," in *Artificial Life II* (SFI Studies in the Sciences of Complexity, vol. X), C. G. Langton, C. Taylor, J. D. Farmer, and S. Rasmussen, Eds. Redwood City, CA: Addison-Wesley, 1992, pp. 371–408.

[44] P. Galley and E. Sanchez, "A hardware implementation of a Tierra processor," Logic Systems Laboratory, Swiss Federal Institute of Technology, Lausanne, Internal Rep. (in French), 1996.

[45] R. Dawkins, *The Blind Watchmaker*. New York: Norton, 1986.

[46] ———, "The evolution of evolvability," in *Artificial Life* (SFI Studies in the Sciences of Complexity, vol. VI), C. G. Langton, Ed. Reading, MA: Addison-Wesley, 1989, pp. 201–220.

[47] D. Mange, M. Goeke, D. Madon, A. Stauffer, G. Tempesti, and S. Durand, "Embryonics: A new family of coarse-grained field-programmable gate arrays with self-repair and self-reproducing properties," in *Toward Evolvable Hardware* (Lecture Notes in Computer Science, vol. 1062), E. Sanchez and M. Tomassini, Eds. Heidelberg: Springer-Verlag,1996, pp. 197–220. (also available as: Dept. Comput. Sci., Swiss Federal Inst. Technol., Lausanne, Switzerland, Tech. Rep. 95/154, Nov. 1995).

[48] "Replicating systems concepts: self-replicating lunar factory and demonstration," *Advanced Automation for Space Missions: Proceedings of the 1980 NASA/ASEE Summer Study* R. A. Freitas, Jr. and W. P. Gilbreath, Eds. Washington, DC: NASA, 1980, chap. 5.
[49] L. Wolpert, *The Triumph of the Embryo*. New York: Oxford Univ. Press, 1991.
[50] J. von Neumann, *Theory of Self-Reproducing Automata*, A. W. Burks. Ed. Urbana, IL: Univ. of Illinois Press, 1966.
[51] U. Pesavento, "An implementation of von Neumann's self-reproducing machine," *Artificial Life*, vol. 2, no. 4, pp. 337–354, 1995.
[52] E. R. Banks, "Universality in cellular automata," in *Proc. IEEE 11th Annual Symposium on Switching and Automata Theory*, Santa Monica, CA, Oct. 1970, pp. 194–215.
[53] A. Burks, Ed., *Essays on Cellular Automata*. Urbana, IL: Univ. Illinois Press, 1970.
[54] E. F. Codd, *Cellular Automata*. New York: Academic, 1968.
[55] J. E. Hopcroft and J. D. Ullman, *Introduction to Automata Theory Languages and Computation*. Redwood City, CA: Addison-Wesley, 1979.
[56] J.-L. Beuchat and J.-O. Haenni, "Von Neumann's 29-state cellular automation: A hardware implementation," submitted for publication.
[57] C. G. Langton, "Self-reproduction in cellular automata," *Physica D*, vol. 10, pp. 135–144, 1984.
[58] J. Byl, "Self-reproduction in small cellular automata," *Physica D*, vol. 34, pp. 295–299, 1989.
[59] J. A. Reggia, S. L. Armentrout, H.-H. Chou, and Y. Peng, "Simple systems that exhibit self-directed replication," *Science*, vol. 259, pp. 1282–1287, Feb. 1993.
[60] K. Morita and K. Imai, "Logical universality and self-reproduction in reversible cellular automata," in *Proc. 1st Int. Conf. Evolvable Systems: From Biology to Hardware (ICES96)* (Lecture Notes in Computer Science). Heidelberg: Springer-Verlag, 1997.
[61] G. Tempesti, "A new self-reproducing cellular automation capable of construction and computation," in *ECAL'95: Third European Conference on Artificial Life* (Lecture Notes in Computer Science, vol. 929), F. Morán, A. Moreno, J. J. Merelo, and P. Chacón, Eds. Heidelberg: Springer-Verlag, 1995, pp. 555–563.
[62] J.-Y. Perrier, M. Sipper, and J. Zahnd, "Toward a viable, self-reproducing universal computer," *Physica D*, vol. 97, pp. 335–352, 1996.
[63] D. Mange and A. Stauffer, "Introduction to embryonics: Toward new self-repairing and self-reproducing hardware based on biological-like properties," in *Artificial Life and Virtual Reality*, N. M. Thalmann and D. Thalmann, Eds. Chichester, UK: Wiley, 1994, pp. 61–72.
[64] D. Mange, E. Sanchez, A. Stauffer, G. Tempesti, S. Durand, P. Marchal, and C. Piguet, "Embryonics: A new methodology for designing field-programmable gate arrays with self-repair and self-reproducing properties," Dept. Comput. Sci., Swiss Federal Inst. Technol., Lausanne, Switzerland, Tech. Rep. 95/152, Oct. 1995.
[65] P. Marchal, C. Piguet, D. Mange, A. Stauffer, and S. Durand, "Embryological development on silicon," in *Artificial Life IV*, R. A. Brooks and P. Maes, Eds. MIT Press: Cambridge, MA, 1994, pp. 365–370.
[66] A. Stauffer, D. Mange, M. Goeke, D. Madon, G. Tempesti, S. Durand, P. Marchal, and C. Piguet, "MICRO TREE: toward a binary decision machine-based FPGA with biological-like properties," in *Proc. Int. Workshop on Logic and Architecture Synthesis*, Institut National Polytechnique de Grenoble, France, 1996.
[67] D. Mange, D. Madon, A. Stauffer, and G. Tempesti, "Von Neumann revisited: A Turning machine with self-repair and self-reproduction properties," Dept. Comput. Sci., Swiss Federal Inst. Technol., Lausanne, Switzerland, Tech. Rep. 96/180, Mar. 1966.
[68] P. Prusinkiewicz and A. Lindenmayer, *The Algorithmic Beauty of Plants*. New York: Springer-Verlgag, 1990.
[69] F. Gruau, "Artificial cellular development in optimization and compilation," in *Toward Evolvable Hardware*, (Lecture Notes in Computer Science, vol. 1062), E. Sanchez and M. Tomassini, Eds. Heidelberg: Springer-Verlag, 1996, pp. 48–75.
[70] H. Kitano, "Designing neural networks by genetic algorithms using graph generation systems," *Complex Syst.*, vol. 4, pp. 461–476, 1990.
[71] J. R. Koza, "Artificial life: Spontaneous emergence of self-replicating and evolutionary self-improving computer programs," in *Artificial Life III* (SFI Studies in the Sciences of Complexity, vol. XVII), C. G. Langton, Ed. Reading, MA: Addison-Wesley, 1994, pp. 225–262.
[72] S. Xanthakis, R. Pajot, and A. Rozz, "Immune system and fault-tolerant computing," in *Evolution artificielle 94*. 1995, Cepadues, cop.
[73] A. Ishiguro, T. Kondo, Y. Watanabe, and Y. Uchikawa, "Immunoid: An immunological approach to decentralized behavior arbitration of autonomous mobile robots," in *Parallel Problem Solving from Nature—PPSN IV* (Lecture Notes in Computer Science, vol. 1141), H.-M. Voigt, W. Ebeling, I. Rechenberg, and H.-P. Schwefel, Eds. Heidelberg: Springer-Verlag, 1996, pp. 666–675.
[74] J. O. Kephart, "A biologically inspired immune system for computers," in *Artificial Life IV*, R. A. Brooks and P. Maes, Eds. Cambridge, MA: MIT Press, 1994, pp. 130–139.
[75] P. Turney, "Myths and legends of the Baldwin effect," in *Proc. 13th Int. Conf. Machine Learning (ICML-96)*, 1996, pp. 135–142.
[76] J. H. Barkow, L. Cosmides, and J. Tooby, Eds., *The Adapted Mind: Evolutionary Psychology and the Generation of Culture*. New York: Oxford Univ. Press, 1992.
[77] S. Pinker, *The Language Instinct: How the Mind Creates Language*. New York: Morrow, 1994.
[78] G. Carpenter and S. Grossberg, "The ART of Adaptive Pattern Recognition by a self-organizing neural network," *IEEE Computer*, pp. 77–88, Mar. 1988.
[79] R. W. Anderson, "Learning and evolution: A quantitative genetics approach," *J. Theoretical Biology*, vol. 175, pp. 89–101, 1995.
[80] ———, "Genetic mechanisms underlying the Baldwin effect are evident in natural antibodies," in *Evolutionary Programming IV: Proceedings of the Fourth Annual Conference on Evolutionary Programming*, J. R. McDonnell, R. G. Reynolds, and D. B. Fogel, Eds. Cambridge, MA: MIT Press, 1995, pp. 547–563.
[81] G. E. Hinton and S. J. Nowlan, "How learning can guide evolution," *Complex Syst.*, vol. 1, pp. 495–502, 1987.
[82] L. E. Atlas and Y. Suzuki, "Digital systems for artificial neural networks," *IEEE Circuits Devices*, pp. 20–24, Nov. 1989.
[83] H. P. Graf and L. D. Jackel, "Analog electronic neural network circuits," *IEEE Circuits Devices*, pp. 44–49, July 1989.
[84] B. Fritzke, "Growing cell structures—a self-organizing network in $k$ dimensions," in *Proc. 1992 Int. Conf. Artificial Neural Networks (ICANN-92)*, I. Aleksander and J. Taylor, Eds. Amsterdam, the Netherlands: North-Holland, 1992.
[85] A. E. Alpaydin, "Neural models of incremental supervised and unsupervised learning," Ph.D. dissertation, Swiss Federal Inst. Technol., Lausanne, 1990.
[86] M. Frean, "The Upstart algorithm: A method for constructing and training feed-forward neural networks," *Neural Computat.*, vol. 2, pp. 198–209, 1990.
[87] C. Cox and W. E. Balnz, "GANGLION: A fast field-programmable gate array implementation of a connectionist classifier," *IEEE J. Solid-State Circuits*, vol. 27, no. 3, pp. 288–299, Mar. 1992.
[88] S. L. Bade and B. L. Hutchings, "FPGA-based stochastic neural networks implementation," presented at IEEE Workshop on FPGA's for Custom Computing Machines, Apr. 1994.
[89] J. G. Eldredge and B. L. Hutchings, "Density enhancement of a neural network using FPGA's and run-time reconfiguration," presented at IEEE Workshop on FPGA's for Custom Computing Machines, Apr. 1994.
[90] A. Perez and E. Sanchez, "FPGA implementation of an adaptable-size neural network," in *Proc. Int. Conf. Artificial Neural Networks (ICANN96)* (Lecture Notes in Computer Science, vol. 1112) C. von der Malsburg, W. von Seelen, J. C. Vorbrüggen, and B. Sendhoff, Eds. Heidelberg: Springer-Verlag, 1996, pp. 383–388.
[91] ———, "Neural networks structure optimization through on-line hardware evolution," in *Proc. World Congr. Neural Networks (WCNN96)*. 1996, INNS (International Neural Networks Society) Press, pp. 1041–1044.
[92] J. M. Moreno, *VLSI Architectures for Evolutive Neural Models*, Ph.D. dissertation, Universitat Politecnica de Catalunya, Barcelona, 1994.
[93] M. Sipper and M. Tomassini, "Co-evolving parallel random number generators," in *Parallel Problem Solving from Nature—PPSN IV* (Lecture Notes in Computer Science, vol. 1141), H.-M. Voigt, W. Ebeling, I. Rechenberg, and H.-P. Schwefel, Eds. Heidelberg: Springer-Verlag, 1996, pp. 950–959.
[94] M. S. Gazzaniga, "Organization of the human brain," *Science*, vol. 245, pp. 947–952, 1989.
[95] B. Happel and J. M. Murre, "Design and evolution of modular neural network architectures," *Neural Networks*, vol. 7, no. 6/7, pp. 985–1004, 1994.
[96] J. Changeux and A. Danchin, "Selective stabilization of developing synapses as a mechanism for the specification of neural networks," *Nature*, vol. 264, pp. 705–712, 1976.
[97] Y. Liu and X. Yao, "Evolutionary design of artificial neural networks with different nodes," in *Proc. IEEE Int. Conf. Evolutionary Computation (ICEC'96)*, 1996, pp. 670–675.

[98] S. Nolfi, D. Parisi, and J. L. Elman, "Learning and evolution in neural networks," *Adaptive Behavior*, vol. 3, no. 1, pp. 5–28, 1994.
[99] X. Yao, "Evolutionary artificial neural networks," *Int. J. Neural Syst.*, vol. 4, no. 3, pp. 203–222, 1993.
[100] D. Ackley and M. Littman, "Interactions between learning and evolution," in *Artificial Life II* (SFI Studies in the Sciences of Complexity, vol. X), C. G. Langton, C. Taylor, J. D. Farmer, and S. Rasmussen, Eds. Redwood City, CA: Addison-Wesley, 1992, pp. 487–509.
[101] D. C. Dennett, *Darwin's Dangerous Idea: Evolution and the Meanings of Life*. New York: Simon & Schuster, 1995.
[102] G. M. Werner and M. G. Dyer, "Evolution of communication in artificial organizms," in *Artificial Life II* (SFI Studies in the Sciences of Complexity, vol. X), C. G. Langton, C. Taylor, J. D. Farmer, and S. Rasmussen, Eds. Redwood City, CA: Addison-Wesley, 1992, pp. 659–687.
[103] B. MacLennan, "Synthetic ethology: An approach to the study of communication," in *Artificial Life II* (SFI Studies in the Sciences of Complexity, vol. X), C. G. Langton, C. Taylor, J. D. Farmer, and S. Rasmussen, Eds. Redwood City, CA: Addison-Wesley, 1992, pp. 631–658
[104] L. Steels, "A self-organizing spatial vocabulary," *Artificial Life*, vol. 2, no. 3, pp. 319–332, 1995.
[105] G. M. Edelman, *Neural Darwinism: The Theory of Neuronal Group Selection*. New York: Basic Books, 1987.
[106] S. R. Quartz and T. J. Sejnowski, "The neural basis of cognitive development: A constructivism manifesto," *Behavioral and Brain Sciences*, submitted for publication.
[107] H. de Garis, "'Cam-Brain' ATR's billion neuron artificial brain project: A three year progress report," in *Proc. IEEE Third Int. Conf. Evolutionary Computation (ICEC'96)*, 1996, pp. 886–891.
[108] F. Gers and H. de Garis, "CAM-Brain: A new model for ATR's cellular automata based artificial brain project," in *Proc. 1st Int. Conf. Evolvable Systems: From Biology to Hardware (ICES96)* (Lecture Notes in Computer Science). Heidelberg: Springer-Verlag, 1997.
[109] A. Roy, S. Govil, and R. Mirand, "A neural network learning theory and a polynomial time RBF algorithm," *IEEE Trans. Neural Networks*, to be published.
[110] J. D. Farmer and A. d'A. Belin, "Artificial life: The coming evolution," in *Artificial Life II* (SFI Studies in the Sciences of Complexity, vol. X), C. G. Langton, C. Taylor, J. D. Farmer, and S. Rasmussen, Eds. Redwood City, CA: Addison-Wesley, 1992, pp. 815–840.
[111] D. G. Stork, B. Jackson, and S. Walker, "Non-optimality via preadaptation in simple neural systems," in *Artificial Life II* (SFI Studies in the Sciences of Complexity, vol. X), C. G. Langton, C. Taylor, J. D. Farmer, and S. Rasmussen, Eds. Redwood City, CA: Addison-Wesley, 1992, pp. 409–42.
[112] A. S. Wu and R. K. Lindsay, "Empirical studies of the genetic algorithm with noncoding segments," *Evolutionary Computation*, vol. 3, no. 2, pp. 121–147, 1995.
[113] T. Cavalier-Smith, *The Evolution of Genome Size*. Chichester, UK: Wiley, 1985.
[114] J. Maynard Smith and E. Szathmáry, *The Major Transitions in Evolution*. Oxford: Freeman, 1995.

**Moshe Sipper** (S'87–M'95) received the B.A. degree in computer science from the Technion–Israel Institute of Technology, Israel, and the M.Sc. and Ph.D. degrees from Tel Aviv University, Israel.

He is a Senior Researcher in the Logic Systems Laboratory at the Swiss Federal Institute of Technology, Lausanne, Switzerland. His chief interests involve the application of biological principles to artificial systems, including evolutionary computation, cellular automata (with an emphasis on evolving cellular machines), bio-inspired systems, evolving hardware, complex adaptive systems, artificial life, and neural networks. He has authored and co-authored several scientific papers in these areas, as well as his recent book *Evolution of Parallel Cellular Machines: The Cellular Programming Approach* (Heidelberg: Springer-Verlag, 1997).

Dr. Sipper was co-organizer of a special session entitled "Toward Evolware," held as part of the IEEE International Conference on Evolutionary Computation (ICEC'97) and is Program Chairman of the Second International Conference on Evolvable Systems: From Biology to Hardware (ICES98), to be held in Lausanne in September 1998.

**Eduardo Sanchez** (M'84) received the diploma in electrical engineering from the Universidad del Valle, Cali, Colombia, in 1975 and the Ph.D. degree from the Swiss Federal Institute of Technology, Lausanne, Switzerland, in 1985.

He is Professor of Computer Science in the Logic Systems Laboratory at the Swiss Federal Institute of Technology, where he is engaged in teaching and research. His chief interests include computer architecture, VLIW processors, reconfigurable logic, and evolvable hardware.

Dr. Sanchez was co-organizer of the inaugural workshop in the field of bio-inspired hardware systems, the proceedings of which are entitled *Toward Evolvable Hardware* (Heidelberg: Springer-Verlag, 1996).

**Daniel Mange** (S'68–M'69) received the M.S. and Ph.D. degrees from the Swiss Federal Institute of Technology, Lausanne, Switzerland.

Since 1969, he has been a Professor at the Swiss Federal Institute of Technology. He held a position as Visiting Professor at the Center for Reliable Computing, Stanford University, Stanford, CA, in 1987. He is Director of the Logic Systems Laboratory, and his chief interests include firmware theory (equivalence and transformation between hardwired systems and programs), cellular automata, artificial life, and embryonics (embryonic electronics). He has authored and co-authored several scientific papers in these areas, as well as the book *Microprogrammed Systems: An Introduction to Firmware Theory* (London: Chapman & Hall, 1992).

Dr. Mange was Program Co-chairman of the First International Conference on Evolvable Systems: From Biology to Hardware (ICES96), held in Tsukuba, Japan, and is General Chairman of the Second International Conference on Evolvable Systems: From Biology to Hardware (ICES98), to be held in Lausanne in September 1998.

**Marco Tomassini** received the diploma in chemistry and physics from the Colegio Nacional Agustin Alvarez, Mendoza, Argentina, and the D.Sc. degree in theoretical chemistry from the University of Perugia, Italy.

He is Professor of Computer Science at the University of Lausanne and a Senior Researcher in the Logic Systems Laboratory at the Swiss Federal Institute of Technology, Lausanne, Switzerland. After completing his studies he conducted research in computational solid state physics and chemistry. During the past decade his interests have centered on parallel and distributed computing, bio-inspired systems, and modeling of physical and economical phenomena.

Dr. Tomassini was organizer of the Sixth Joint EPS-APS International Conference on Physics Computing and co-organizer of the inaugural workshop in the field of bio-inspired hardware systems, the proceedings of which are entitled *Toward Evolvable Hardware* (Heidelberg: Springer-Verlag, 1996).

**Andrés Pérez-Uribe** received the diploma in electrical engineering from Universidad del Valle, Cali, Colombia, in 1993. From 1994 to 1996 he held a Swiss government fellowship and is currently a Ph.D. candidate in the Department of Computer Science at the Swiss Federal Institute of Technology in Lausanne.

Since 1994 he has been with the Logic Systems Laboratory, Swiss Federal Institute of Technology, working on the digital implementation of neural networks with adaptable topologies, in collaboration with the Centre Suisse d'Electronique et de Microtechnique SA (CSEM). His research interests include neural networks, field-programmable devices, neurocontrollers, and bio-inspired systems.

**André Stauffer** (S'68–M'69) received the diploma in electrical engineering and the Ph.D. degree from the Swiss Federal Institute of Technology, Lausanne, Switzerland.

He is a Senior Lecturer in the Department of Computer Science at the Swiss Federal Institute of Technology. He spent one year as a Visiting Scientist at the IBM T. J. Watson Research Center in Yorktown Heights, NY. He also collaborates with the Centre Suisse d'Electronique et de Microtechnique SA in Neuchâtel, Switzerland. In addition to digital design, his research interests include circuit reconfiguration and bio-inspired systems.

Dr. Stauffer was co-organizer of a special session entitled "Toward Evolware," held as part of the IEEE International Conference on Evolutionary Computation (ICEC'97).