Chapter 13

Towards synthetic evolution of nanostructures

Hod Lipson

Mechanical & Aerospace Engineering and Computing & Information Science
Cornell University, Ithaca NY 14853, USA
E-mail: Hod.Lipson@cornell.edu

Abstract

This article begins to consider key ingredients necessary to apply non-biological evolutionary processes at the nano scale. In such processes, large numbers of building blocks spontaneously self-organize into new irregular forms that were not directly predetermined by a designer, but rather form solutions to a given functional requirements. Such structures will be able to adapt their configuration and behavior in response to changing requirements and changing conditions, ultimately leading to a form of evolutionary materials. The motivation for looking into self-organizing phenomena like evolution at the nano-scale is both for finding new ways do design structures, and to synthetically recreate and examine the evolutionary processes that gave rise to primordial life. This article describes some lessons learnt from applying evolutionary processes at macro-level robotic structures, and postulates as to key ingredients that will be necessary to complete an evolutionary cycle at the nano scale.

1. Introduction 342
2. Evolution of structures 342
3. Scaling 343
4. Modularity 346
5. An indirect method for inducing modularity 348
6. Towards physical implementation 349
7. Conclusions 351
References 352

1. Introduction

The understanding of nano-structure processes is taking place at several levels simultaneously. Some research is focused on understanding the physics of single building-block units, such as the electrical, mechanical and chemical properties of carbon nanotubes and DNA strands. At the next level there is interest in the interface physics that allows single building blocks to connect to each other in a controllable way to make junctions, structures and machines. At a still higher level, we are interested in self-assembly processes that allow very large numbers of building blocks to form semi-regular structures according to a predetermined set of rules. Finally, we may now start considering techniques that will allow large numbers of building blocks to adaptively self-organize into irregular forms that were not directly predetermined by a designer, but rather form solutions to some given functional requirements. Such structures will be able to adapt their internal configuration and behavior in response to changing requirements and changing conditions, ultimately leading to a form of *evolutionary materials.*

The motivation for looking into self-organizing phenomena like evolution at the nano-scale is twofold. First, the difficulty in manipulation and design of small-scale structures calls for new techniques of design, construction and testing. Evolutionary mechanisms offer an alternative paradigm for automating this cycle while avoiding low-level discrete manipulation. In particular, with proper setup, some self-organizing phenomena can allow for closing the design-test loop by harnessing the massively parallel nature of the nanoscale substrate to search the design space. The second reason for looking into self-organizing processes at the nanoscale is the potential to synthetically recreate and examine physical evolutionary processes similar to those that gave rise to primordial life. Such experimentation will allow testing of some basic hypotheses on the emergence of biological life, and possibly even recreating very simple artificial life forms in a synthetic substrate.

This brief paper does not present any experimental results achieved at the nano-structure scale. Instead, I will describe evolutionary robotics experiments carried out in scale-less simulation and verified at the physical macro level. Although many aspects of macro-scale physics to not apply to nanoscale systems, some of the phenomena, especially those concerned with the emergence of complexity, are broadly applicable to self organizing systems in general. Lessons learnt from evolution of structures in simulation may thus shed some light on possible paths towards application of evolutionary processes at the nano scale, and specifically the roles of modularity and hierarchy in achieving structures with complex functionality. Finally, I will conclude with some general ideas about possible ingredients necessary for implementation of evolutionary processes in the physical nanoscale substrate.

2. Evolution of structures

Genetic algorithms (GAs) – a subset of evolutionary computation involving mutation and crossover in a population of fixed length bit strings - have been applied for several decades in many engineering problems as an optimization technique for a

fixed set of parameters. But open-ended systems, in which the process is allowed to add more and more building blocks and parameters, seem particularly adequate to describe evolutionary processes that might occur at the nano scale. Open-ended evolutionary design systems have been demonstrated for a variety of simple design problems, including structures, mechanisms, software, optics, robotics, control, and many others (for overviews see, for example, Refs. [1,2]). Yet these accomplishments remain simple compared to what teams of human engineers can design and what nature has produced. The evolutionary design approach is often criticized as scaling badly when challenged with design requirements of higher complexity.

In a set of experiments, we investigated the possibilities of evolving locomotive structures out of bars, linear actuators and neurons. We then used commercial rapid prototyping techniques to transfer the evolved machines into reality to test their physical viability. Details of these experiments and their results can be found in earlier reports [3]. Here I will briefly overview the findings and then discuss their scaling properties.

In these experiments, the physics model considered combinations of bars, actuators and neurons. The bars connect to each other through free joints, neurons can connect to other neurons through synaptic connections, and neurons can connect to bars. In the latter case, the length of the bar becomes governed by the output of the neuron, essentially making it a linear actuator. This model was chosen because it allows for a large variety of structure: Trusses can form arbitrary rigid, flexible and articulated structures as well as multiple detached structures. Bars connected with free joints can form configuration acting as revolute, linear and planar joints at various levels of hierarchy. Similarly, sigmoidal neurons can connect to create arbitrary control architectures such as feed-forward and recurrent nets, state machines and multiple independent brains. A schematic illustration of a possible architecture is shown in Fig. 1.

The goal was to evolve machines (structure and control) that could locomote over an horizontal plane. Starting with a population of 200 blank machines that were initially comprised of zero bars, zero actuators and zero neurons, we conducted evolution in simulation. The fitness of a machine was determined by its locomotion ability: the net distance its center of mass moved on an infinite plane in a fixed duration. The process iteratively selected fitter machines, created offspring by randomly adding, modifying and removing building blocks, and replaced them into the population. The process was typically carried out for over 600 generations. Some results are shown in Fig. 2.

3. Scaling

While there are still many poorly understood factors that determine the success of evolutionary design - such as starting conditions, variation operators, primitive building blocks and fidelity of simulation - one problem is that the design space is exponentially large, because there is an exponentially increasing number of ways a linearly increasing set of components can be assembled. Consequently, evolutionary

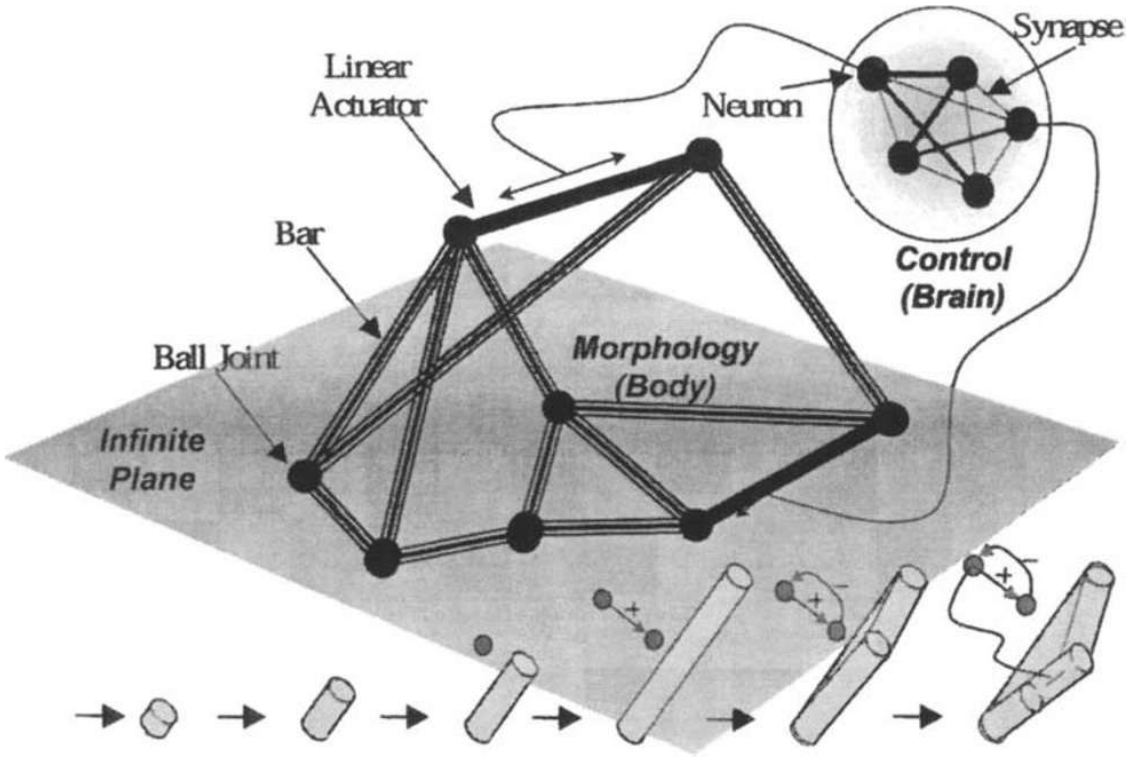


Fig. 1: Schematic illustration of an evolvable robot, containing only linear bar/actuators and control neurons. Some bar architectures form rigid substructures.

approaches that operate on direct encodings quickly become intractable because of combinatorial complexity.

This is evident in our own work: Fig. 3 shows a typical progress of fitness of evolved machines as function of generations. The abscissa represents evolutionary time (generations), the ordinate measures fitness (net movement on a horizontal plane) and each point in the scatter plot represents one candidate robot. In general, after an initial period of drift, with zero fitness, we observe rapid growth followed by a logarithmic slowdown in progress, characterized by longer and longer durations between successive step-improvements in the fitness. This real time lingering is amplified by the fact that evaluation time or duration of a generation (in simulation or in physical reality) also increases as solutions become more complex. However, note that because of the stochastic nature of the process, it is hard to determine definitely whether progress has actually halted, and improvements may still occur after long periods of apparent stagnation (Fig. 3c, 3d).

Looking at the kind of structures that result after leaving the system running for extended periods of time (weeks, in practical terms), we see that the kind of structures that evolve exhibit high internal coupling (Fig. 4). While these machines do have a slightly higher fitness than their predecessors, they are complex to the point where it is difficult for them to continue evolving efficiently. In other words, their *evolvability* is impaired.

From engineering design principles and from evidence in nature, we know that

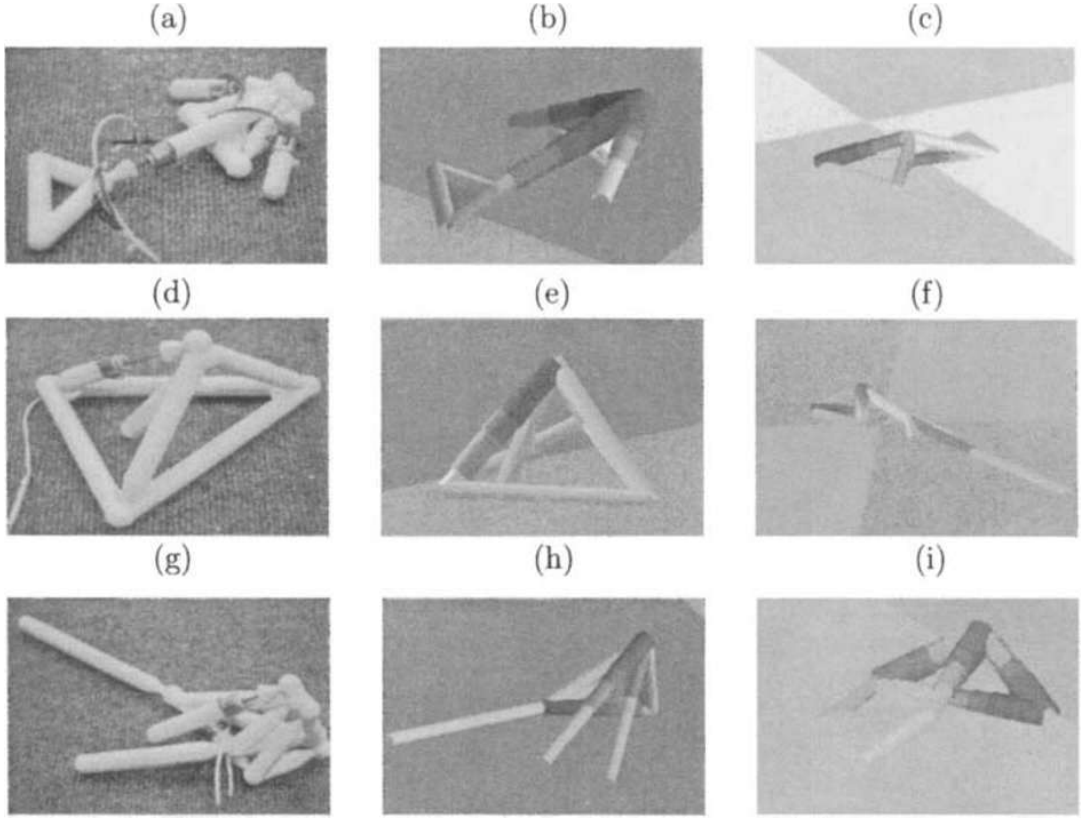


Fig. 2: Some results of the evolutionary process (from [3]): (a,b) This surprisingly symmetric machine uses a 7-neuron network to drive the center actuator in perfect anti-phase with the two synchronized side limb actuators. While the upper two limbs push, the central body is retracted, and vice versa. (d,e) A tetrahedral mechanism that produces hinge-like motion and advances by pushing the central bar against the floor. (g,h) This mechanism has an elevated body, from which it pushes an actuator down directly onto the floor to create ratcheting motion. It has a few redundant bars dragged on the floor, which might be contributing to its stability. (c,f,i) other converged machines, all produces with the same parameter settings. These machines perform in reality is the same way they perform in simulation. Motion videos of these robots and others can be viewed at http://www.demo.cs.brandeis.edu/golem

one of the keys to maintaining evolvability is the use of architectural principles of regularity, modularity and hierarchy. Below I will briefly discuss some of these new approaches and their implementation in evolutionary computation. However, these methods require a genotype-phenotype separation that is not necessarily available in non-biological physical nano-scale evolutionary processes. And so I will propose an alternative weak approach that may induce modularity without requiring a direct genotype-phenotype representation.

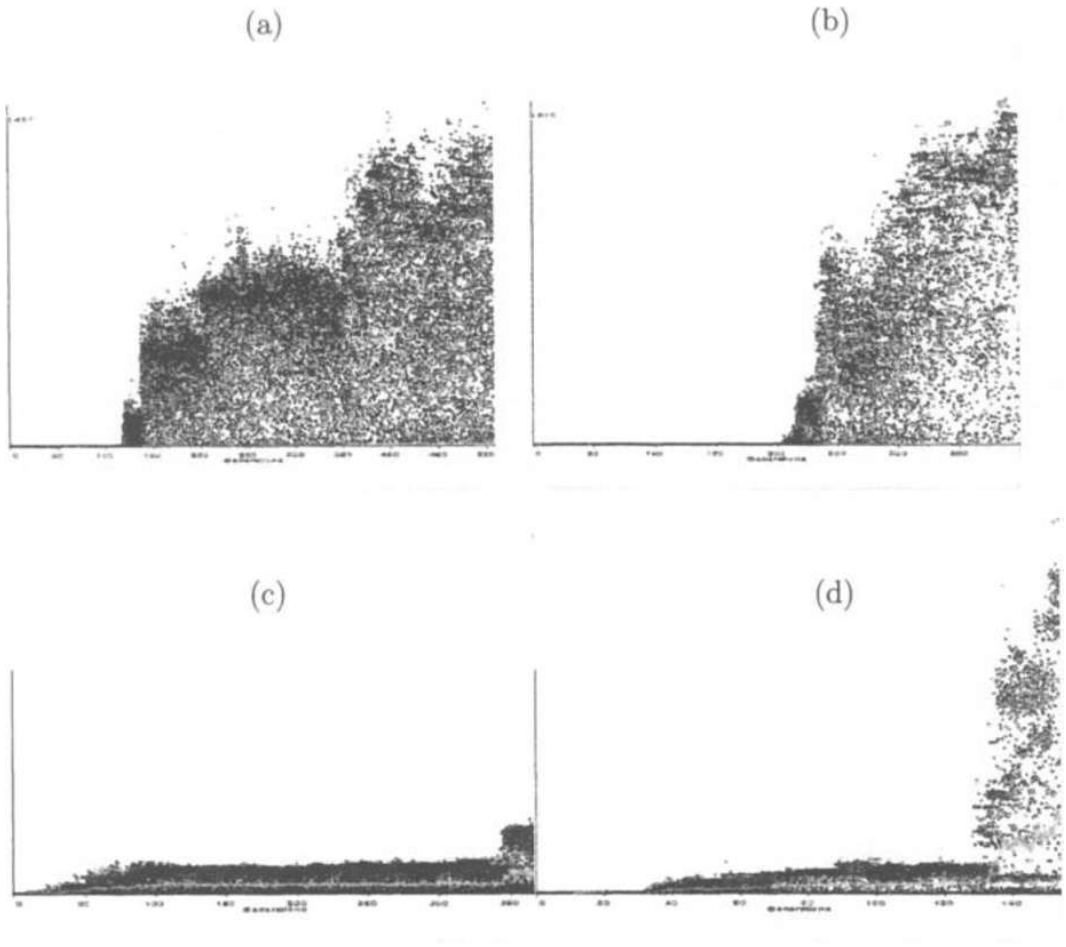


Fig. 3: Progress of a typical evolutionary design run. The abscissa represents evolutionary time (generations) and the ordinate measures fitness. Each point in the scatter plot represents one candidate design. In general, a logarithmic slowdown in progress can be observed, characterized by longer and longer durations between successive step-jumps in the fitness (a,b). Occasionally, however, progress is made after long periods of stagnation (c,d).

4. Modularity

It has long been recognized that architectures that exhibit functional separation into modules are more robust and amenable to design and adaptation [4,5]. Modularity creates a separation that reduces the amount of coupling between internal and external changes, allowing evolution to rearrange inputs to modules without changing their intrinsic behaviors, and so to reuse modules as high-level building blocks. In nature this idea is supported by theoretical arguments such as that proteins are difficult to evolve once they are participating in many different interactions, and by observations of phenomena such as tight coordination of the expression of groups of genes functioning in a common process, as well as evidence of modules appearing as a robust segmentation mechanism for handling developmental noise. Conversely,

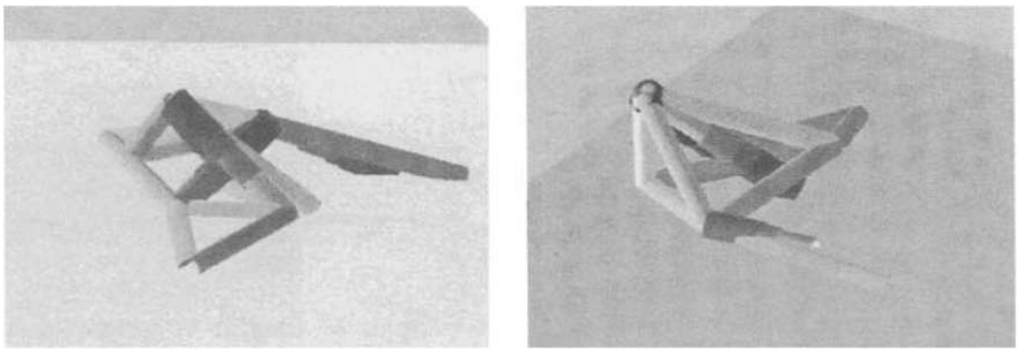


Fig. 4: Machines evolved after very large number of generations of mutations may have a slightly higher fitness, but generally highly coupled internally.

there is evidence that proteins which interact with many other proteins, such as histones, actin and tubulin, have changed very little during evolution. In artificial systems modularity is critical too: Herbert Simon [6] noted, in his famous "Tempus and Hora" fable, that the evolution of complex forms from simple elements depends critically on the numbers and distribution of potential stable intermediate forms. Modularity has also been recognized as a primary facilitating characteristic of system engineering (e.g., Ref. [7]), economics [8], and named as one of the principles of design.

Perhaps one of the first arguments to the importance of building blocks was put forward by Holland [9] through the building block hypothesis. The building block hypothesis suggests that genetic algorithms are able to identify low-order schemata (low-level modules) with above-average fitness and combine them, through crossover operators, to produce high-order schemata (high-level modules), and continue doing so recursively. However, it has been shown that this powerful property relies on proper diversity of the population and, more importantly, on genetic linkage - the proximity along the genotype of genes that encode for related function. In absence of these assumptions genetic algorithms will not scale well [10]. It is clear that when we address synthesis problems where the initial building blocks are scattered around, as in physical nanoscale setup, there is no way to guarantee genetic linkage. Moreover, when there is no genotype at all, the term is undefined.

One approach to avoiding reliance on genetic linkage is based on Genetic Programming [11] principles, which deal with partial specifications directly and manipulate them in coherent tree structures. This concept is enhanced through a mechanism for automatically defining functions (ADF) that maintains partial solutions (modules) in separate populations. However, direct manipulation of modules may be difficult at the physical nanoscale.

Another alternative that allows production of modularity without assuming genetic linkage is the generative approach. Here a structure is not specified directly, but is specified through an evolving coding that in turn generates the structure. Like

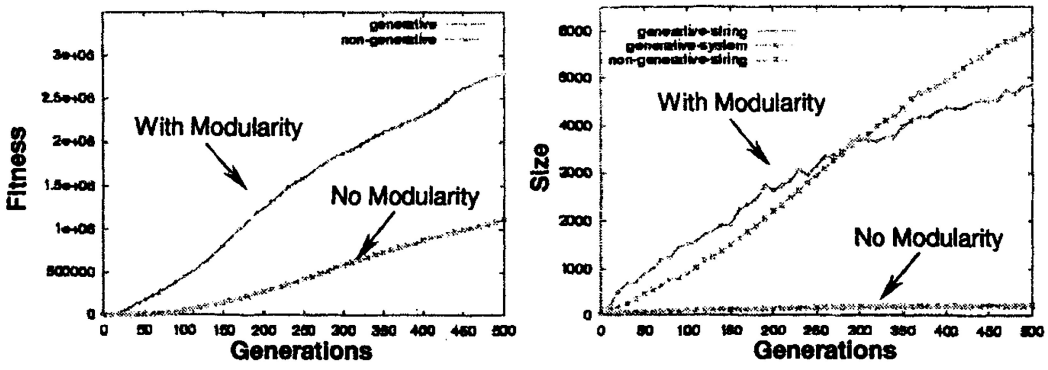


Fig. 5: Comparison of fitness (left) and complexity (right) as function of generation, in a generative versus non-generative substrate, for evolution of locomoting structures [12].

a structured computer program, a generative specification can allow the definition of re-usable sub-procedures allowing the design system to use loops and recursion to produce large regular structures by reiterating a small set of commands. The DNA, combined with a developmental process, is an example of this kind of process. Generative systems can promote both modularity (separation of function) and regularity (reuse), through this indirect process. In a preliminary set of experiments [12], we used Lindenmayer systems (L-systems) as the generative encoding to be evolved: Evolution produces L-system programs, those programs in turn generate construction sequences that in turn construct robots. Figure 5 shows a comparison of fitness and complexity (measured by description length) as function of generation, in a generative versus non-generative substrate, for evolution of static structures. Although this does not prove scalable behavior, it provides support for the notion that modularity and regularity can significantly accelerate progress.

A more recent approach to enhancing modularity and hierarchy is the Symbiogenic Evolutionary Adaptation Model [13]. This model works with a population of partially specified solutions and tests them in context of each other to combine them into higher-level partial solutions. This combinative process relies on co-existence of the subcomponents for evaluation and Pareto-dominance criterion to decide on transition between hierarchy levels. It has been shown to solve efficiently complex test problems that do not assume genetic linkage yet have an exponential number of local minima, and are therefore more closely applicable to physical nanoscale evolution.

5. An indirect method for inducing modularity

The physical nanoscale evolution substrate is characterized by three properties that do not allow direct application of standard modularity enhancing methods. First, there is no linkage, i.e., the building blocks are scattered around randomly and we cannot assume that building blocks that happen to lie in proximity are functionally related. Thus, methods that rely on linkage (like standard Genetic Algorithms) cannot be applied. Second, there is no genotype (only a phenotype), so methods that require a genome (like generative approaches) cannot be directly applied, unless they use an intermediate representation like a living cell. Third, it is difficult to directly

manipulate modules, so that methods that rely on module manipulation, like Genetic Programming, would also be difficult to implement. We seek an indirect method for enhancing modularity that does not rely on a particular representation.

In a recent study [14], we have chosen to examine the spontaneous emergence of modularity using a simple and abstract model of an adaptive system as a transformation of a set of resources into a set of arbitrary functional requirements for survival. We model a natural evolving system as a transfer matrix **A** that is required to transform a vector of given environmental resources **E** into some vector **F** representing a set of arbitrary functional requirements: F = A×E (see Fig. 6 a). The figure of merit $f(\mathbf{A})$ of the design candidate **A** is how well it satisfies the above vector equation, given a vector of requirements **F** and a vector of resources **E**. This figure of merit (or *fitness*, in evolutionary terms) can be quantified as the magnitude of deviation $|\mathbf{F} - \mathbf{AE}|$. Based on the definitions above, we quantify modularity as inversely proportional to the amount of coupling $C(A)$ in the system. We simulated a simple evolutionary process where a population of **A**'s was evolved for a given pair of **F** and **E**. We observed the dependency of the average coupling $C(A)$ on the rate of change $d\mathbf{E}/dt$ of the environment resources.

The main results are summarized in a plot of internal coupling versus environment change rate, shown in Fig. 6 b. Each point in the plot is averaged for 100 experiments, each of 20,000 generations of a population of 200 matrices. This plot shows a clear linear-log correlation that can be quantified as an empirical modularity law that relates the amount of coupling in a system C to the rate of change of the problem:

$$C = -k \log \frac{dE}{dt} + C_0$$

where k and C_0 are constants that are dependent on the mutation bias, the amount of interaction between the system and the environment, and other specifics of the substrate. This experiment suggests that modularity arises spontaneously in evolutionary systems in response to a changing environment, and that the amount of modular separation is logarithmically proportional to the rate of change of the environment. This quantitative model can shed light on the evolution of modularity in nature, and predicts that modular architectures would appear in correlation with high environmental change rates. In the nanoscale evolution context, these results also suggest that modularity can be induced within physical systems by evolving in a changing environment. This approach does not assume linkage, is not based on a particular representation, and does not require direct manipulation of modules.

6. Towards physical implementation

Evolutionary processes require a set of founding blocks that can be varied and replicated in a fitness-dependent way. The existence of founding building blocks whose interfaces can be preprogrammed has already been established and has been demonstrated to assemble into large regular structures like DNA crystals [15]. The bonding energy of these structures is at the same approximate of level of ambient

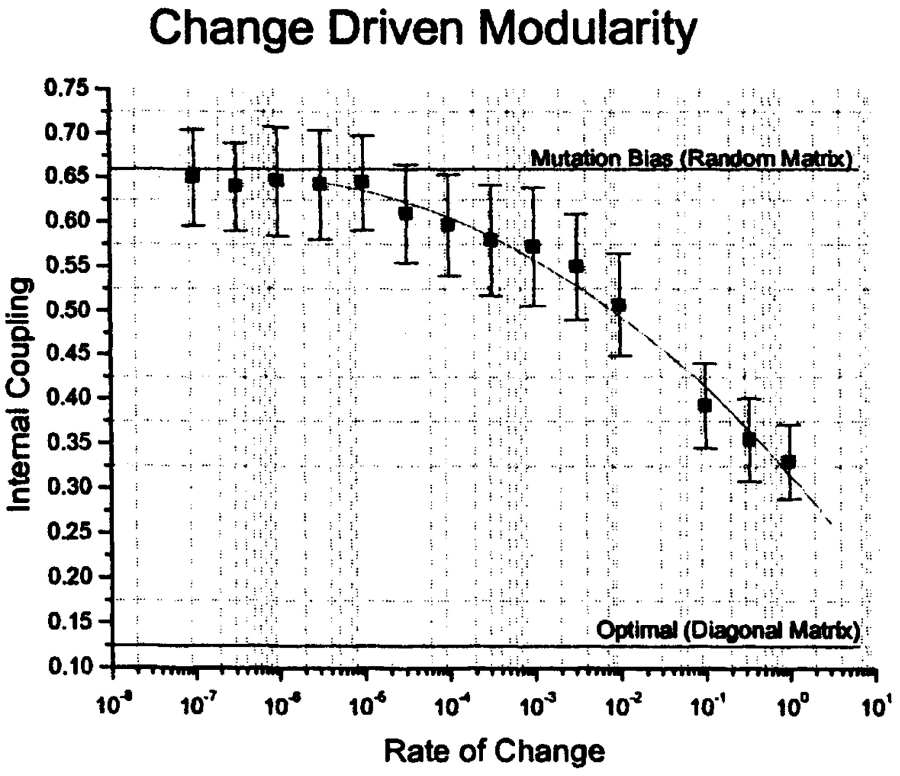
$$\begin{bmatrix} -1 \\ +1 \\ -1 \\ -1 \\ +1 \\ +1 \\ +1 \\ -1 \end{bmatrix} \stackrel{?}{=} \begin{bmatrix} +1 & 0 & -1 & -1 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -11 & +1 & 0 \\ -1 & 0 & +1 & 0 & 0 & 0 & +1 & 0 \\ 0 & -1 & 0 & 0 & +1 & 0 & 0 & 0 \\ -1 & 0 & 0 & +1 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & +1 & -1 & 0 & 0 & -1 & 0 \\ +1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \end{bmatrix} \times \begin{bmatrix} +1 \\ -1 \\ +1 \\ +1 \\ -1 \\ +1 \\ +1 \\ -1 \end{bmatrix}$$

F — An arbitrary set of functional requirements or survival requirements

A — A design candidate or individual trying to transform building blocks to meet requirements

E — An arbitrary set of building blocks or resources in environment

(a)



(b)

Fig. 6: Environment driven modularity: (top) An abstract general design problem, where a matrix **A** is required to transform a set of building blocks E to meet a set of requirements **F**. Many solutions exist, and, (bottom) Internal coupling versus environment change rate shows a linear-log correlation.

energy of the system (kT), and so 'design variations' are relatively easy to induce. While in most cases this source of variation and error is considered a problem, for evolution it is in fact a source of innovation. The trick is inversely linking the rate of variation with the design goal, or the 'fitness' of the structure, so that good designs will be stable, while poor design will be subject to more variation. This fitness-dependent-variation link needs to be established in a non-centralized way in order to exploit the massively parallel nature of the nanoscale substrate. For example, if we are interested in evolving machines that can move, say, we could expose the substrate to a moving energy field that would transfer more energy into static structures than to moving structure. This energy would thus induce more variation in lower fitness structures. Another example might be evolving structure to sustain a load. Here, we need to design markers that attach to DNA sequences and absorb more of a radiating field when the structure is under high stress.

7. Conclusions

There are still many key aspects of evolutionary process that require a physical implementation before an entire evolutionary cycle can be completed in nanoscale reality. While there is much effort and significant results in establishing good building blocks whose interface can be controlled, we still need to address ways of producing fitness dependent variation and replication. Even when we are able to do this, experiments in other domains indicate that the level of complexity we would be able to achieve would still be limited. Early results in DNA computing that did not consider scaling were deemed impractical when applied to more complex problems. Similarly, in search of practical implementation of evolutionary processes we must consider ways to promote modularity and hierarchy if we expect these methods to scale to higher complexities. One way to achieve modularity is through the use of biological agents like cells, who have already 'solved' much of the representation and replication issues. However, to induce a fully synthetic evolutionary process we need to do so without relying on intermediate representations and on direct manipulation, and some ideas towards these goals are presented here.

References

[1] P. J. Bentley, (Ed.), *Evolutionary Design by Computers* (Morgan Kaufman 1999).

[2] P. Husbands and J. A. Meyer, *Evolutionary Robotics* (Springer Verlag 1998).

[3] H. Lipson and J. B. Pollack, Nature **406**, 974 (2000).

[4] L. H. Hartwell, J. H. Hopfield, S. Leibler, and A. W. Murray, Nature **402**, C47 (1999).

[5] G. P. Wagner, American Zoologist, **36**, 36 (1996).

[6] H. A. Simon, *The Sciences of the Artificial* (MIT Press 1996), 3rd edition, (The fable of Tempus and Hora).

[7] C. C. Huang and A. Kusiak, IEEE Transactions on Systems, Man, and Cybernetics, Part A, **28**, 66 (1998).

[8] R. N. Langlois, Journal of Economic Behavior and Organization, in press (2001).

[9] J. Holland, *Adaptation in natural and artificial systems* (University of Michigan Press 1975).

[10] R. A. Watson and J. B. Pollack, GECCO-99 Late Breaking Papers, 292 (1999).

[11] J. Koza, *Genetic Programming* (MIT Press 1992).

[12] G. S. Hornby, H. Lipson, J. B. Pollack, IEEE International Conference on Robotics and Automation (2001).

[13] R. A. Watson andJ. B. Pollack, Parallel problem solving from nature 2000, PPSN VI (2000).

[14] H. Lipson, J. B. Pollack, and N. P. Suh, Proceedings of DETC'01 2001 ASME Design Engineering Technical Conferences, September 9-12, 2001, Pittsburgh, Pennsylvania, USA (2001).

[15] E. Winfree, F. Liu, L. A. Wenzler, N. C. Seeman, Nature **394**, 539 (1998).