Synthetic Biocomputation: the possible and the actual

Ricard V. Solé[1,2,3] and Javier Macia[1,2]

[1] ICREA-Complex Systems Lab, Universitat Pompeu Fabra, PRBB-CEXS. Dr Aiguader 80, 08003 Barcelona, Spain
[3] Santa Fe Institute, 1399 Hyde Park Road, 87501 Santa Fe, New Mexico, USA
[2] Institut de Biologia Evolutiva, UPF-CSIC, Barcelona
ricard.sole@upf.edu, javier.macia@upf.edu

Abstract

Computation is defining trait of biological systems and a broad framework that captures the complex adaptive nature of molecules, cells and organisms. Computation is also at the core of the genotype-phenotype mapping, since it provides a natural framework to define function in a self-consistent way. The study of existing biological systems (from signalling cascades to ant colonies or brains) as well as the evolution of synthetic in silico networks performing computations reveals a number of nontrivial patterns of organization, sometimes in clear conflict with standard view of engineering or optimization. In spite of our increasing knowledge, there is a lack of a theoretical framework where computation and its possible forms is integrated within a general picture. Synthetic biology provides a new avenue where engineered molecular circuits can be implemented to perform non-standard computations. Here we review recent advances in the domain of multicellular synthetic computing and suggest a potential morphospace of computational systems including both standard and non-standard approximations.

Introduction

Computation in nature is a fascinating and yet difficult topic. Biological systems perform computations as they gather information and process it in order to respond to environmental cues. Computation is in fact one formal way of capturing functionality in a well defined fashion (1), (2). Computation has also become a key aspect within the emergent field of synthetic biology (for a recent review, see (19)). This field allows to construct completely new molecular and cellular structures able to perform artificial computations (3).

Cells can be engineered in order to behave as autonomous, potentially programmable computing devices. These biocomputing devices would be able to perform complex tasks and designed for a wide range of applications, including bioremediation, food production or biomedicine (4). How to make these systems reusable and scalable is a major problem, but new approaches involving non-standard forms of computing have been able to overcome some key difficulties (5). They define novel ways of computing using living matter and suggest potential scenarios to outline a general framework to unify the landscape of computational structures, both in the natural and artificial realms.

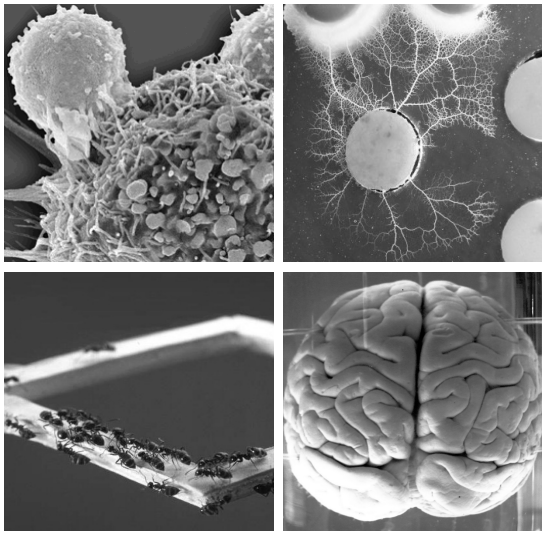

Figure 1: Computation occurs in natural systems in many different systems and spanning multiple scales. This include immune networks, social insect colonies, brains or some social amobeae.

In order to use computation as a unifying framework where biological complexity and its evolutionary dynamics can be suitably integrated, some formalism is needed. One possibility is to consider classical models of computation. Turing's formalization of computations in terms of machines with a number of internal states provides a powerful framework where -in principle- any potential form of computation could be described (6). The fact that some particular macromolecular systems, such as ribosomes act pretty much as Turing-like nanomachines (reading a "tape" defined by the messenger RNA, creating an output chain of aminoacids and starting and ending the process by means of detecting given sequences) seems to support this picture. Such avenue has been successfully taken by some researchers (7) proving the viability of making molecular computations close to finite automata. However, as pointed out by Melanie Mitchell (8) there is a range of biological systems, from immune net-

works to ant colonies or even plants, where computations occur and yet seem to escape from being fully captured by classical, Turing-like formal approaches to computation.

The special features shown by information-processing systems in biology have been recognized for decades. Many of them have to do with special ways of treating given computational tasks in a parallel way and using the internal dynamical features characteristic of each system. Task allocation in ants, for example, can be favoured in some cases by means of colony-level oscillations which seem in principle inappropriate for dealing with colony needs. Simple models of ant dynamics based on a neuron-like mapping between ant states and formal neurons have bee very useful in this context. In particular, it has been shown that oscillations actually favour an optimal task fulfilment that is not possible if a constant, average activity level were at work (9), see also (10).

Similarly, other properties exhibited by complex biological machines strongly depart from standard engineering-based principles. One such principle is the robust behavior based on redundancy. Here two identical components of the system making the same function can replace each other in case of failure. Redundancy is thus the intuitive (although sometimes expensive) solution to the problem of failure. However, it has been shown that in many cases (may be in most cases) robust behavior is not obtained from redundant structures. Instead, it seems to be a consequence of so called degeneracy (11), (12), (13). It can be defined as the capacity of elements of a given system that are structurally different to perform the same function or yield the same output. This ubiquitous feature appears to be present in many diferent systems and scales. Modeling in silico evolved circuits performing computations under selection for robust behavior (14) reveal that robustness is achieved through degeneracy, but the underlying mechanistic explanation escapes from our intuition. Degeneracy implies a novel concept beyond standard engineering, suggesting that new forms of thinking might be required.

How can we go beyond the limits imposed by real systems, which are the result of evolution and might be difficult to fully characterize? Similarly, how can we test existing theories and try novel ones if they are sometimes difficult to compare with their real counterparts? The field of synthetic biology seems to provide the best scenario for designing novel computational systems in vivo whereas non-standard forms of computation are used as alternatives to engineering-inspired metaphors. Here we present some of these results and suggest a potential framework to define a space of computational designs that includes existing natural and artificial systems as well as engineered, artificial ones.

Logic gates from gene circuits

One way of creating synthetic biological circuits performing predefined logic operations is based on engineering genetic

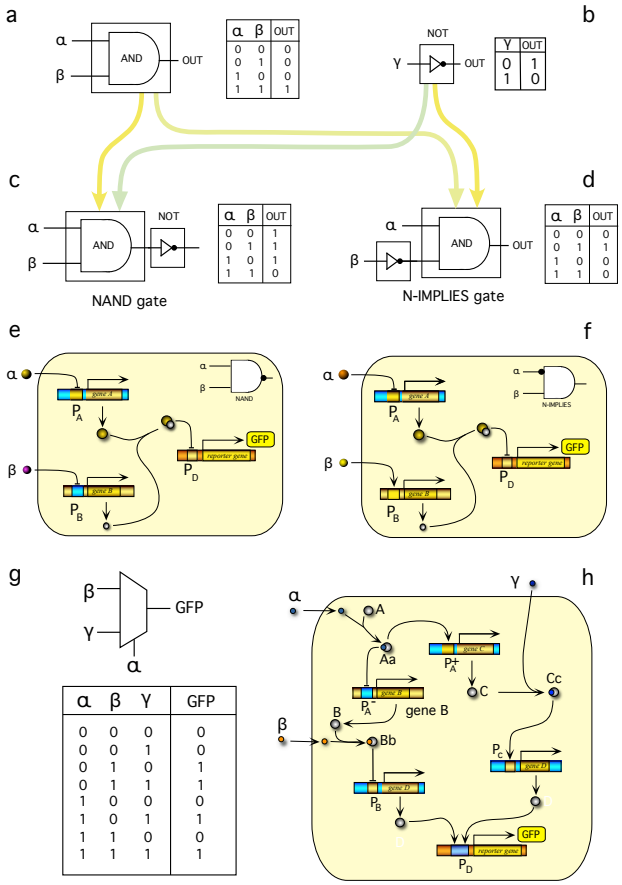

Figure 2: Logic gates and switches of different types can be obtained by engineering cellular and/or molecular systems. Examples would include (a) the AND and (b) the NOT gates, from which a NAND gate (c) or a N-IMPLIES gate (d) can be obtained through combination. In (e) and (f) we illustrate these two examples through a hypothetic gene regulatory system (the inset pictures are the compact representation of the gates).

regulatory systems. In figure (2) we show some examples of logic gates that can be implemented by using available genetic components and their interactions. Such circuits are obtained by means of standard genetic engineering techniques and the components can actually come from different, completely unrelated species, which can mix together genes from viruses, bacteria or mammals. Typically these engineered circuits are built within plasmids, i. e. closed chains of DNA defining genetic information physically separated from the chromosomal DNA. A different strategy involves using appropriate gene. In figure 2a we illustrate this by means of two basic examples and their genetic counterparts (other implementations are also possible). In our example (e) the NAND gate is obtained by using a molecular complex formed by two different proteins which repress the

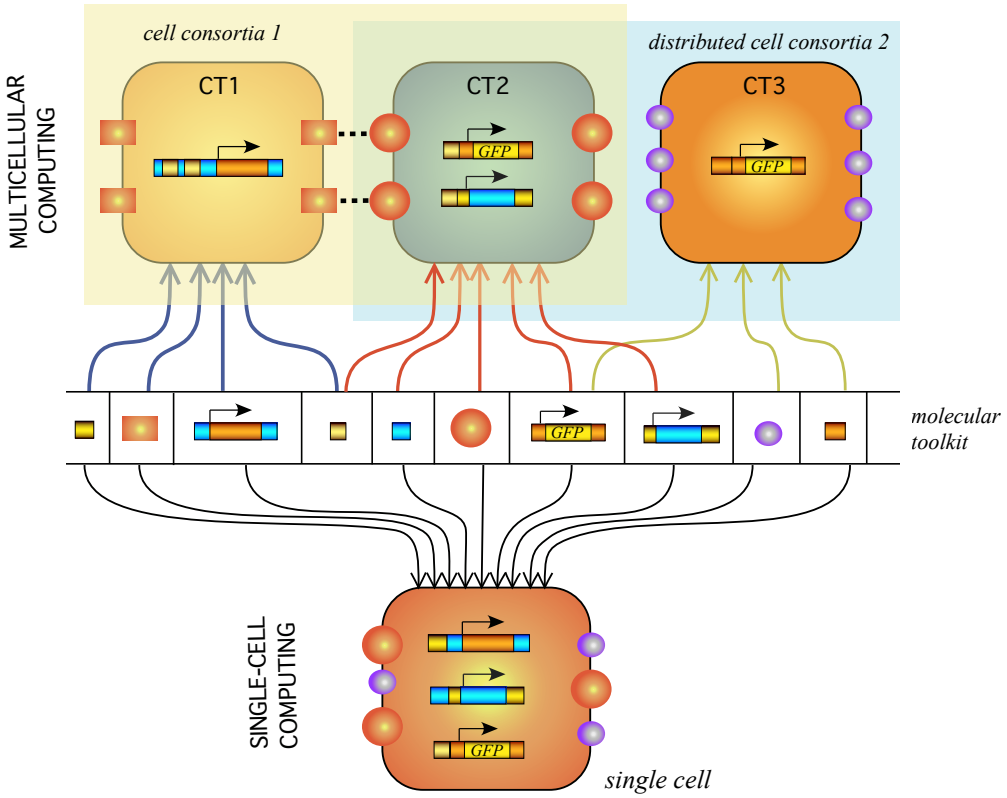

Figure 3: A genetic toolkit (central line) can be used to engineer cellular computations. The toolkit might include all sorts of regulatory elements and reporters. The standard approach is putting together within a single cell, where all regulations take place. Alternatively, a library of different engineered cells can be created, thus defining a cellular consortia (top diagrams).

expression of a so called reporter gene (here GFP=green fluorescent protein) which generates, when activated, a fluorescent signal.

These examples illustrate the standard approach of electronic design based on combinatorial logic. In principle, every circuits could be designed in this way. However, a major difficulty emerges here: in electronics, every wire is defined in terms of a conducting piece of material, which is always the same. When dealing with cellular engineered systems, where molecules share the same medium where they are mixed, identity becomes a problem. In a cell, every wire needs to be a different molecule to properly connect different elements or cells. Because the liquid nature of the medium where computations need to occur, the spatial insulation of wires that is assumed in electronics is no longer satisfied. As a consequence, each wire needs to be implemented by using a different molecular carrier and the chemical diversity of constructs rapidly grows. This is illustrated in figure 1g-h by the so called multiplexor, widely used in electronic designs. This is a 3-input, one-output system where a given signal ”selects” one of the two inputs. In principle a synthetic genetic network implementing a MUX circuit can be designed (an example shown in figure 1h) using a single cell implementation. Although such circuit can be constructed (15) it is a hard task, with no hope of being re-used as part of a larger system (as it occurs in electronics).

To sum up, the combinatorial approach can lead to a nightmare when dealing with an experimental design, since the properties of each carrier and how it interacts with other can be very different and difficult to predict. Additionally, one goal of the field is to have engineered systems capable of extensive reuse of available parts in such a way that a LEGO-like system is at work. Both premises are basic requirements for reaching the computational complexity for achieving autonomous machines able to make decisions in a biological context. Only recently a general approach, based on engineering several cell types, has been successfully obtained.

Cellular consortia: division of labor

One way of dealing with the wiring problem is considering alternative ways of avoiding the mixing of molecular carriers that seems inevitable within the cell cytoplasm. Spatial segregation of the basic components provides one easy way of dealing with computation avoiding molecular mixing. Although the explicit use of spatial locations is one possibility

(see for example (16) a simpler scenario involves using cellular consortia, namely a population of cells having different types of engineering designs. In such scenario, a library of different cell types, each one having a different subset of genetic components, is build out of a collection of molecular components. This is schematically displayed in figure 3, where we show three different ways of combining them within single or multicellular constructs.

Here different cell "types" are indicated as CT1, CT2, etc. A standard consortium (top left) is obtained by splitting some of the elements from the toolkit between cells. Communication is then also introduced, so that a sender and a receiver cell are usually designed, although feedbacks are also introduced in most designs. A reporter cell is present (here CT2) which will (1) or will not (0) express a target molecule. This type of consortium has been used in many different contexts. In particular, using two cell types it was possible to artificially recreate predator-prey systems (Lotka-Volterra dynamics), mutualistic ensembles (hypercycle-like systems) or parasitic organizations. Extensions of these include multispecies ecosystems where different groups of cells belonging to different kingdoms are involved (17). Once again, however, the resulting synthetic cells are hard to reuse to obtain other types of computations. An alternative approach requires breaking some predefined rules.

In any standard circuit design, the truth table defines the input-output relation between incoming sets of signals and the resulting outputs. The outputs are placed in given locations of the circuit and it makes sense that this is the case. Let us limit ourselves here to a single-output system. That means that there is an output unit where the final result of the information processing is released. What happens if we free ourselves from such (rather reasonable) assumption? The view of a computational device as being implemented by a circuit that clearly differentiates between input, processing and output units seems too obvious to replace it by some other paradigm. But there is actually one solution that emerges from not forcing that assumption to be true. Instead, more than a cell type is able to respond as output element.

Distributed computation

Here we introduce our basic model approach to synthetic computation. We will use a Boolean approximation, thus confining our approximation to the digital domain. Our state space will be described by a set $\Sigma = \{0, 1\}$. Although this is in principle a limitation, many relevant cellular computations seem to take place by means of genetic switches. Such switches effectively define binary states with low and high levels of gene expression. A given functionality will be described as an input string **I**, namely an element of

$$\Sigma^N = \underbrace{\{0,1\} \times \ldots \times \{0,1\}}_{n} \tag{1}$$

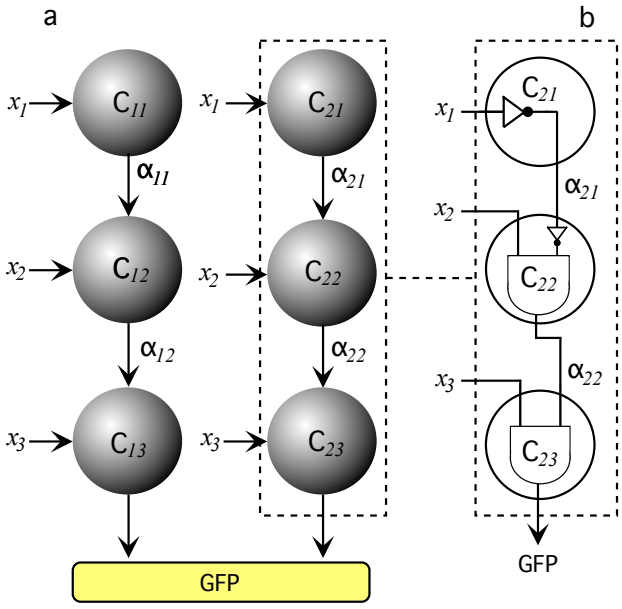

Figure 4: The general architecture used here to generate our circuits using distributed computation, as defined in the text. A feed-forward structure (a) is assumed as the basic scaffold, with a number of input signals that affect separated arrays of communicating cells (gray spheres). Each cell implements a given logic function (b). All columns end up in a reporter cell, but several reporter cells can be present, since each column is seggregated from others, thus removing potential cross-talks.

It will indicate, in our framework, a string of absent (0) or present (1) chemical signals.

The functional trait to be implemented is formally defined as a Boolean function ϕ_i with N input signals and a single output. Formaly, this reads:

$$\phi_i : \Sigma^N \longrightarrow \Sigma \tag{2}$$

Two particularly relevant subsets of Boolean functions are the one input-one output gates, i. e. the set $\mathcal{G}^{(1,1)} = \{NOT, Id\}$ (the negation and identity functions, respectively) and the 2^4 two-input logic gates defining the set $\mathcal{G}^{(2,1)} = \{g_j\}$ where g_j is a maping

$$g_j : \Sigma^2 \longrightarrow \Sigma \tag{3}$$

(represented by a simple table). Standard functions include OR, AND and their "inverse", i. e. NOR and NAND.

Our approach to a general design of complex computational circuits (5) is based on two general assumptions, to be translated into a basic circuit design (fig 4). First, we limit ourselves to a feed-forward network where each node is one type of engineered cell from the library. Each link means the existence of a molecular connection, i. e. a diffusible

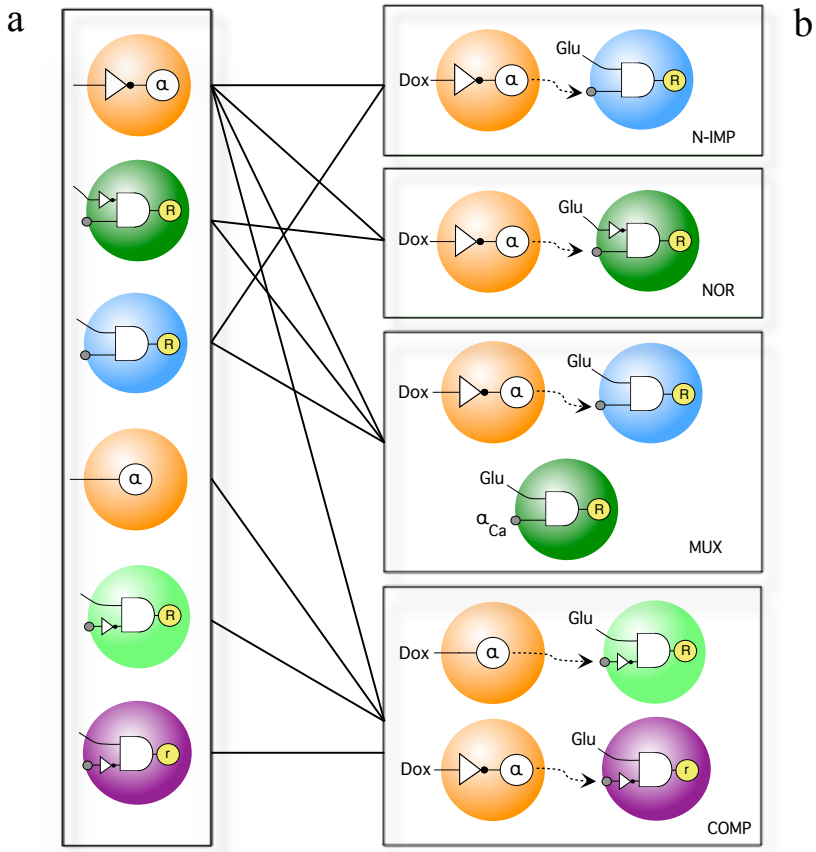

Figure 5: Combinatorial design of multicellular consortia using our distributed computation method. Here in (a) we sketch some basic engineered cell types (implemented on yeast cells) and several examples (b) of combinations performing given functions. Many other circuits can also be constructed using this library or small extensions of it.

wire molecule. Secondly, several cell types can incorporate the gene responsible for the output molecule (GFP). Once a given Boolean table is chosen, an evolutionary algorithm is applied to the basic wiring structure, which explores the landscape of potential networks implementing the desired function (3). The algorithm searches over the space of basic functions, wiring configurations and other constraints. Once a given network is found, standard rules of circuit minimization are applied in order to obtain the minimal circuit solving the problem by means of distributed computation.

What are the results of this method? Along with this evolutionary algorithm, the theoretical analysis demonstrates that it is possible to minimize the number of required cells and wires using the distributed output assumption combined with a small library of cells implementing only the AND and the inverted Implies gates (N-IMPLIES). Despite this combination of gates are not usually used in circuit designs they define a functional complete set, i.e. any arbitrary Boolean function can be implemented only combining this two gates. In some embodiments, these gates can be simplified and replaced by the IDENTITY and the NOT gates respectively allowing for a circuit simplification. Furthermore, the wiring pattern of connections is restricted, i.e. different circuits can involve different number of cells and wires but all cells only respond to an external input and to single diffusible molecule acting as a wire according with the specific logic function implemented, i.e. AND or N-IMPLIES, independently on the circuit complexity.

Using yeast cells as the model organism to implement our cell library (following the theoretical predictions) it was possible to construct, by combining different cell types, all kinds of simple gates (figure 5) but also complex circuits. As an illustration of the enormous simplification of circuit complexity that is derived from our approach, in figure 5 we

can see that the MUX circuit can be obtained by combining three cells from the library (it can also be done with only two). Similarly, much more complex circuits, such as a binary adder (figure 5) was also obtained. As we can see from the two complex circuits, the assumption of distributed computation makes possible to actually split the circuit in different segregated (and thus disconnected) parts. Once again, this is in deep contrast with the standard view of electronics.

A final result concerns the predicted types of cell-cell interactions that is predicted by the evolutionary algorithm. Using the MUX circuit as the basic reference, we run the algorithm in such a way that many different circuits were obtained, all consistently implementing the multiplexer. In figure 6 we display a graph summarizing two relevant pieces of information. The nodes are the basic gates implemented by individual cells. Their size here is proportional to their frequency in the evolved circuits. We can see that there are wide differences between different logic components. Secondly, the weighted links between different gates indicate how frequently two given gates appeared connected within a given solution. The resulting network illustrates once again the nonstandard character of our solutions. The first lesson is that, although it is known that NOR and NAND gates could be in principle used as the single logic elements to implement any logic circuit (18) this solution is largely ignored by the algorithm. Secondly, the N-IMPLIES function, which was successfully used in (5) seems to be a key component in most solutions. Since the N-IMPLIES gate is not a standard component in electronics but seems to be very important here, this suggests that some design principles used in synthetic biology might need to be revisited.

The potential power of distributed computation as described above is illustrated by noticing that even a small number of engineered cell types makes possible to create hundreds of synthetic circuits (5) and thus a huge potential array of functions. Adding wires makes the combinatorial power of the system to rapidly increase in orders of magnitude the number of potential circuits, which are easily achieved thanks to the enormous capacity for tinkering and combination.

Discussion

Synthetic biology has been rapidly gaining relevance and potential as novel techniques are getting incorporated to the field and new applications start to emerge (**?**), (20), (21). Our view of the area in terms of computation is simply a way of addressing the combinatorial potential of functional circuits in a very broad way. Such view allows to properly address some of the key problems in the field, namely wiring constraints and real combinatorial design. Our recent work indicates that by removing the assumption of specified output units, by allowing the output to be distributed over multiple cell types, low-wiring, combinatorial circuits can be obtained.

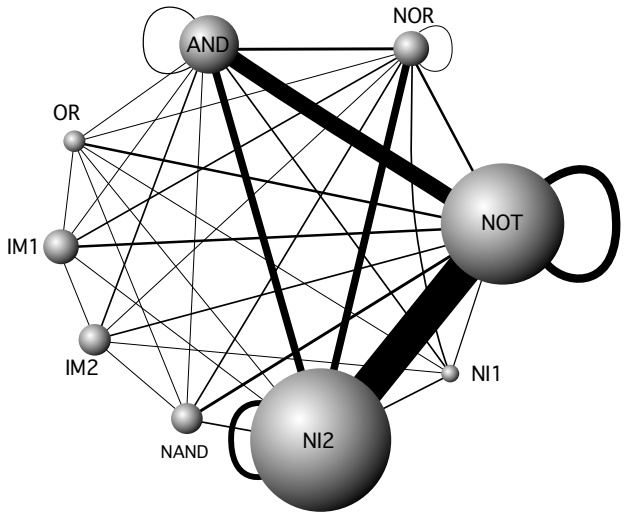

Figure 6: The weighted network obtained for many different evaluations of the evolutionary algorithm searching for multicellular MUX networks.

The previous results are encouraging in two different ways. On the one hand, given the truly combinatorial potential of the method, hundreds of possible synthetic designs can now be constructed. The method allows to predict possible ways of building minimal circuits and thus adapt the required result to experimental constraints. But it also opens an interesting framework to approach more general questions. Our method shows that an unexpected way of solving computational problems can be obtained.

The resulting solutions are counterintuitive and reveal an alternative form of actually achieving the right computation through cellular consortia that can be disconnected into several pieces. Moreover, the results might be more general. For convenience, we have presented our work in terms of cellular consortia, where the basic, spatially defined units are cells. But it might well be the case that other scenarios, such as sub-cellular structures, also fit within our framework. Different cellular compartments could in principle perform parts of the computational processing required to implement a given function in a distributed manner. Since biology tends to make possible everything that can be imagined under reasonable terms, we predict that the kind of computations presented here are likely to be found in living systems.

It is also interesting to notice that reliable computation at low wiring cost has been achieved through a method where autonomous parts emerge as part of the solution. Given a function to be implemented, the architecture of the resulting design involves different parts contributing to the overall computation but essentially independent. This result suggests that the evolved circuits might actually display a high degree of robustness, and preliminary results seem to con-

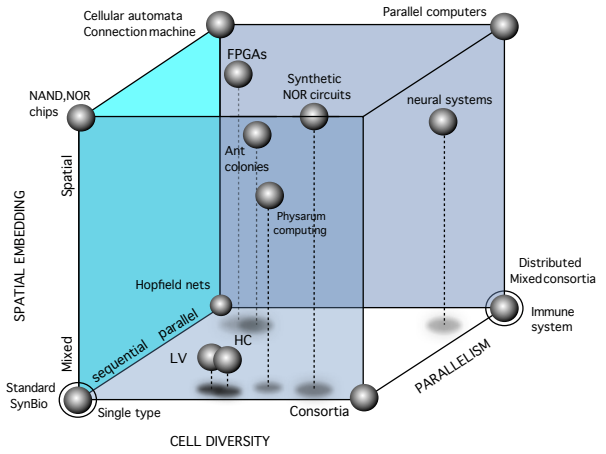

Figure 7: A toy space of computations where the three axes include the presence of spatial segregation, degree of multicellularity and to what extent the computation is sequential or parallel. Several well known examples are located at roughly representive locations. Here LV= Lotka-Volterra synthetic ecosystem, HC=mutualistic synthetic system, NAND, NOR chips: small chips constituted by only NAND and NOR gates, widely used as basic building blocks in many electronic designs; FPGAs: field programmable arrays.

firm this point.

Finally, our results also introduce an additional layer of complexity within biological computation. If we define an imaginary, qualitative space of computational structures where the number of different cells and the presence of space define two axes, a third one would be the relative importance of distributed computation as defined here (in terms of the output). A tentative (and by no means exhaustive) picture of this space is provided in figure 7. In this space, we have allocated different known systems that differ in their computational power and how it works. Our distributed computation approach defines, a corner of this diagram (encircled, right sphere). The three axes are intended to capture three relevant features of computational systems. These are: (a) degree of parallelism, (b) diversity of units involved (cells for our engineered systems but can be ant castes or electronic components in other contexts) and (c) spatial embedding, meaning how relevant is the spatial distribution of the agents while performing computations.

This is a largely unexplored space, where spatial degrees of freedom can help to further simplify our implementation and simultaneously increase our combinatorial power (unpublished results). Moreover, it is possible to show that a limit case of our implementation, where all cells in the consortia are actually disconnected among them, can successfully be implemented too. Many open questions emerge from this work, but it also provides an elegant and promising scenario where many relevant questions will be testable, including potentially unexplored forms of computation, their robustness and evolution.

Acknowledgements

We would like to thank the members of the Complex Systems Lab as well as to F. Posas, L. de Nadal, N. Conde and S. Regot for useful discussions. RVS also thanks his colleagues S. Kauffman, M. Mitchell, D. Krakauer, C. Moore and JF Sebastian for interesting comments. This work has been supported by grants from the Ministerio de Ciencia y Innovación (BIO2009-07762) and the James McDonell Foundation; the CELLCOMPUT (FP6) project, The Santa Fe Institute. Special thanks are given to the Fundación Marcelino Botin (FMB).

References

Amos, M. (2004). Cellular Computing. *Oxford University Press*, New York.

Regev, A. and Shapiro, E. (2002). Cellular abstractions: Cells as computation. *Nature*, 419:343.

Macia,J., Posas, F. and Sole, R.V. (2011). Synthetic biological computers. *Trends in Biotechnology*, In press.

Benenson, Y. (2009). Biocomputers: from test tube to live cells. *Mol. BioSyst*, 5:675-685.

Regot, S., Macia, J., Conde, N., Furukawa, K., Kjellen, J., Peeters, T., Hohmann, S., de Nadal, E., Posas, F. and Sole R.V. (2011). Distributed Biological Computation with Multicellular Engineered Networks. *Nature*, 469:207-211

Turing A.M. (1937). On computable numbers, with an application to the Entscheidungsproblem. *Proc. London Math. Soc.*, 42:230-265

Benenson, Y., Paz-Elizar, T., adar, R., Keinan, E., Livneh, Z. and Shapiro, E. (2011). Programmable and autonomous computing machine made of biomolecules. *Nature*, 414:430-434.

Mitchel, M. (2011). Ubiquity symposium: Biological Computation. *Volume 2011 Issue February*, ACM New York, USA.

Delgado, J., and Solé, R.V. (2000). Self-synchronization and taks fulfilment in ant colonies. *J. Theor. Biol.*, 205:433-411

Solé, R.V. and Delgado, J. (1996). Universal Computation in Fluid Neural Networks. *Complexity*, 2:49-56.

Tononi, G., Sporns, O. and Edelman, G.M. (1999). Measures of degeneracy and redundancy in biological networks. *Proc. Natl. Acad. Sci. USA*, 96:3257-3262.

Edelman, G.M. and Gally, J.A. (2001). Degeneracy and complexity in biological systems. *Proc. Natl. Acad. Sci. USA*, 98:13763-13768.

Wagner, A. (2005). Robustness and evolvability in living systems. *Princeton U. Press*. Princeton.

Macia, J. and Sole, R.V. (2009). Distibuted robustness in cellular networks: insights from evolved diital circuits. *J. R. Soc. Interface*, 442:259-264.

Moon, T.S., Clarke, E.J., Tamsir, A., Clark, R.M., Eanes, M., Kortemme, T. and Voigt C.A. (2011). Construction of a genetic multiplexer to toggle between chemosensory pathways in *Escherichia Coli*. *J Mol Biol*, 406:215-27.

Tamsir, A., Tabor, J.J. and Voigt, C.A. (2011) Robust multicellular computing using genetically encoded nor gates and chemical 'wires'. *Nature*, 469: 212-5.

Weber, W., Daoud-El Baba, M. and Fussenegger, M. (2007). Synthetic ecosystems based on airborne inter- and intrakingdom communication. *Proc. Natl. Acad. Sci. USA*, 104:10435-10440.

Enderton, H. (2001). A mathematical instroduction to loic. Harcourt Academic Press.

Purnick, P.E. and Weiss, R. (2009). The second wave of synthetic biology: from modules to systems. *Nature Reviews Molecular Cell Biology*, 10:410-422.

Abelson, H., Allen, D., Coore, D., Hanson, C., Homsy, G., Knight, T.F., Nagpal, R., Rauch, E., Sussman, G.J. and Weiss R. (2000). Amorphous computing. *Communications ACM*, Volume 43, Number 5.

Weiss, R., Basu, S., Hooshangi, S., Kalmbach, A., Karig, D., Mehreja, R. and Netravali, I. (2003). Genetic circuit building blocks for Cellular Computation, Communications, and Signal Processing, *Natural Comput.*, 2:47-84.