How can we test for artificial life? Complexity and Computation in Nature

Hector Zenil

hector.zenil@ki.se

November 7, 2013

ShangAI Lectures on Natural and Artificial Intelligence

Karolinska Institutet

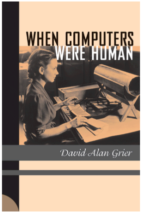
We all believe we know how computers look like

And how they definitely don't look like. Right?

(Yellow slime mold: a single cell! with thousands of nuclei. They are formed when individual flagellated cells swarm together and fuse.)

Human computers

But they did use to look like this!

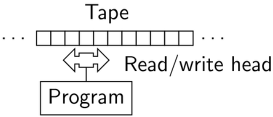


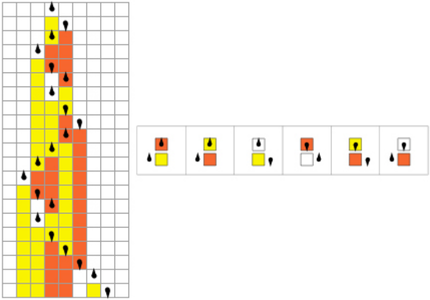
A *computer* at work, c. 1940.

Oxford English Dictionary: "A person employed to make calculations in an observatory, in surveying, etc."

When Computers Were Human by Grier, David Alan (Nov 1, 2013)
http://www.computerhistory.org/revolution/calculators/1/65/2209

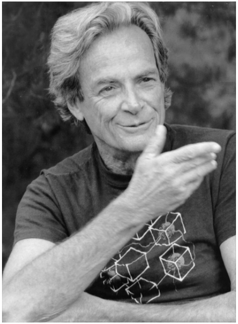
Turing's computing model





Other computationally equivalent models: Post Tag systems, Church lambda calculus, Kleene recursive functions.

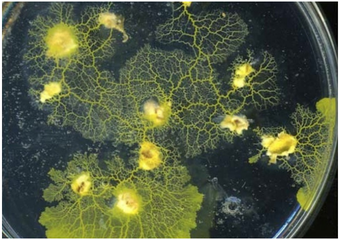
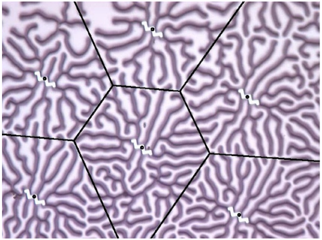
Richard Feynman



I don't believe in computer science.... To me science is the study of the behaviour of nature...

Natural computing

Some computer scientists are trying to compute with completely different substrates.

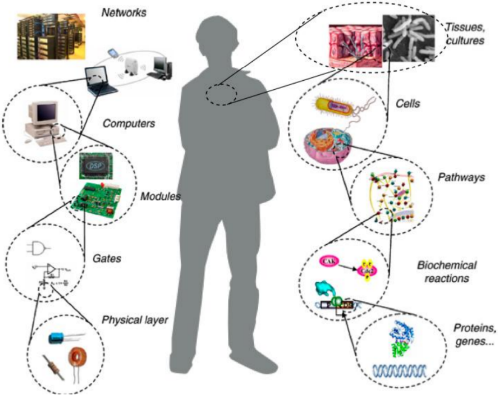
 

Physarum (slime mold) (left) and Electrified liquid crystals (right) Slime moulds are especially adept at finding optimal solutions to transportation network problems, such as paths and even inspiring a solution to the (NP complete) travel salesman person problem and Voronoi partitions.

WIRED

Computers Made Out of DNA, Slime and Other Strange Stuff (04.02.13)

What is a computer? A Digital? Electronic? Quantum device?



(e.g. circadian clock in plants and animals: computing time in a 24hr period)

Can a computer simulate the universe?

In fact, Feynman asked what kind of computer could simulate the universe...

Classical computers cannot simulate the universe because the full description of quantum physics requires too many variables. If one particle is described by 2 variables, for example, then to simulate 100 particles need 2^{100} (a 1 followed by 30 zeroes).

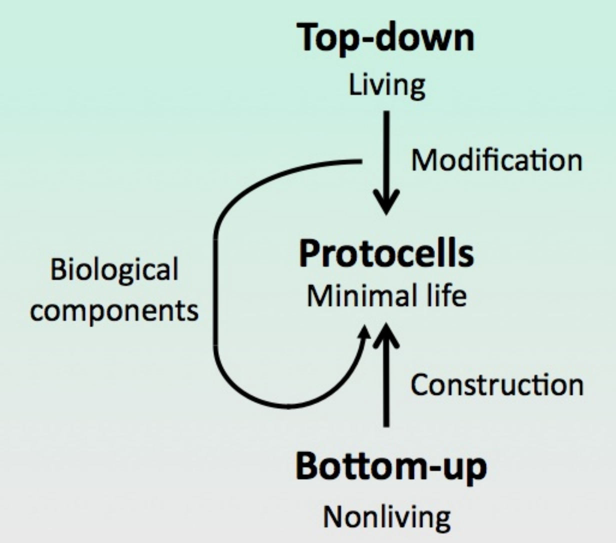
Quantum computers: qubits. They keep up with the *world capacity*. (and can solve faster some classical computations)

Whether quantum computers are actually possible and scalable is yet unknown. Nor it is known whether a more fundamental classical substratum gives rise to quantum phenomena. But we can ask about natural computation, even if not quantum mechanical.

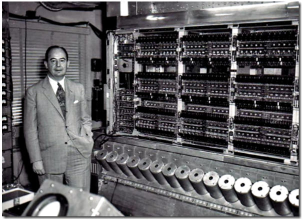
Natural computation and computation in nature

Fungi “fairy” circles and lava stone hexagons.

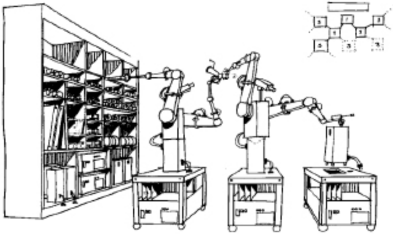
Reengineering vs creating life



John von Neumann

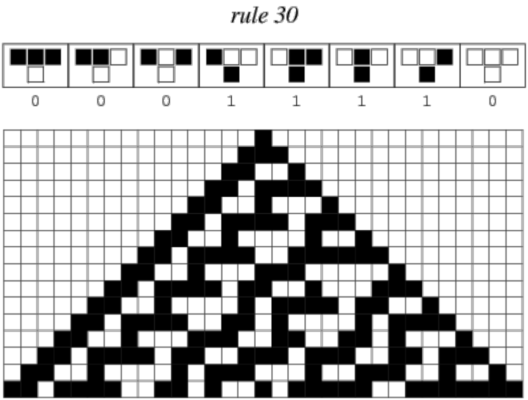


J. von Neuman's self-replication interest (1950s).

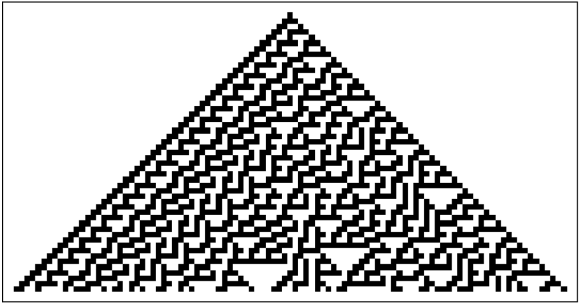


von Neumann (with S. Ulam) started the field of Cellular Automata

Cellular Automata



Rule 30 long run

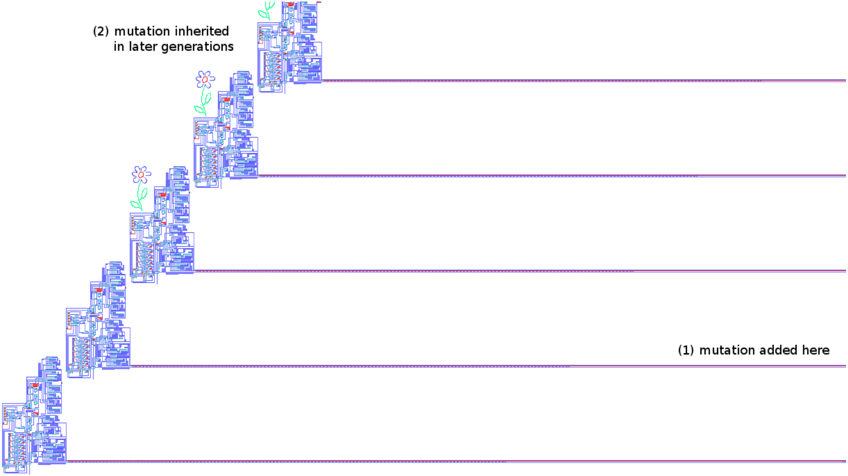


Rule 30 operating in nature?

Image: Zenil, 2012.

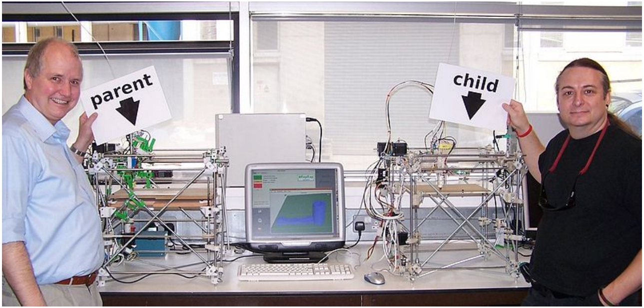
von Neumann's universal constructor (cont.)

J. von Neumann's universal constructor (claimed to have 28 states).



RepRap

The RepRap machine (University of Bath , UK) prints all its components, it (almost) self-replicates (it does not assemble the pieces).

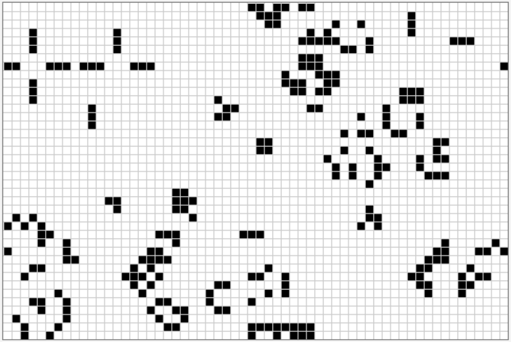


In computer science a self-reproducing computer program is a computer program that, when executed, outputs its own code. This is also called a *quine*. Here is an example program in the Python programming language:

```
a='a=%r;print a%%a';print a%a
```

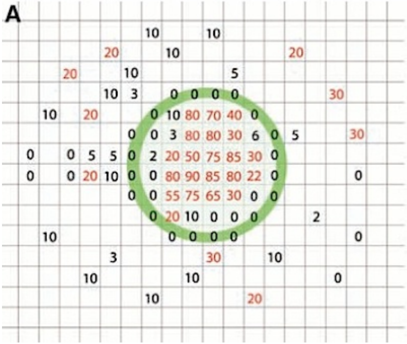
Conway's Game of Life

Mimics many properties of life (movement, persistence, interaction, evolution).



It is capable of Turing universality!

Cellular automata as models of nature

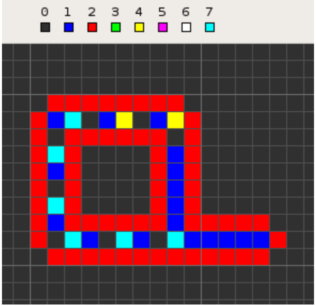


"Fairy" circles explained with cellular automata.

Image: Norbert Juergens and Science (2013).

Life as self-replication?

Langton's loop shows that self-replication does not require Turing universality.

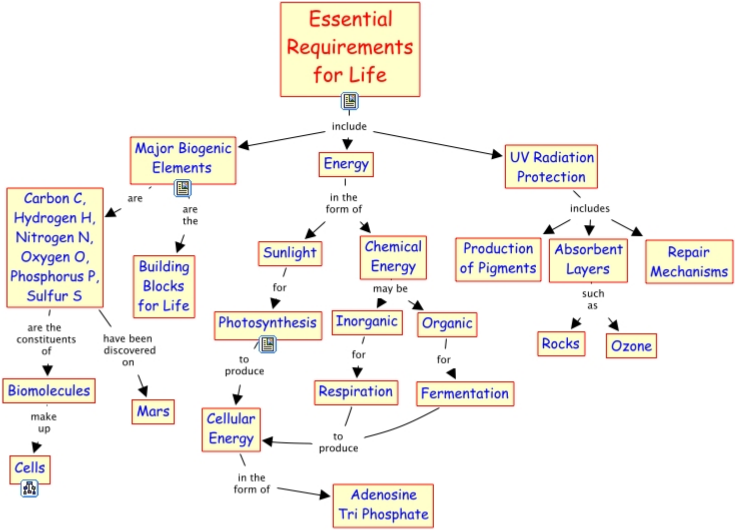


Crystals...

However, self-replication is not enough by itself to characterise life:

Crystals self-replicate too!

Essential requirements for life?



Information, computation and life

An informational definition (G.F. Joyce, *Bit by Bit: The Darwinian Basis of Life*, 2012) of life:

> *A (genetic) system that contains more bits than the number that were required to initiate its operation.* (added parenthesis)

A computational view (John Hopfield, J. Theor. Biol. 171, 53-60, 1994):

> *All biological systems perform some kind of computation. Computation is inherent to adaptive systems and makes biology different from physics.*

Leroy Cronin (TEDx conference, 2011) proposes that:

> *Life is anything that can undergo evolution (survival of the fittest). Matter that can evolve is alive.*

Borderline case: Virus evolve (not exactly by their own though) yet they are only matter (encapsulated genetic material).

Do things *really* compute?

There is a current debate around the concept of natural computationalism. Do things and living systems compute?

Stephen Wolfram (A New Kind of Science, (2002)):

> *There is a simple rule for our universe.*

"Rule" here means a ***Turing computable*** rule.

Seth Lloyd (Programming the Universe, (2007)):

> *The universe computes itself.*

"Computes" here means ***quantum computes***.

Hillary Putnam's "Multiple realizability" criticism to computationalism (functionalism).

Chalmer's "Does a Rock Implement Every Finite-State Automaton?"

I want a measure to associate it to (and try answer) these claims/questions.

Algorithmic information theory to the rescue?

Which string looks more random?

(a) 1111111111111111111111111111111111111111
(b) 0011010011010010110111010010100010111010
(c) 0101010101010101010101010101010101010101

Definition

$$K_U(s) = min\{|p|, U(p) = s\} \quad (1)$$

Compressibility and algorithmic randomness

A string with low Kolmogorov complexity is c-compressible if $|p| + c = |s|$. A string is random if $K(s) \approx |s|$.

[Kolmogorov (1965); Chaitin (1966)]

Example of an evaluation of K

The string (b) 01010101...01 is not algorithmic random (or has low K complexity) because it can be produced by the following program:

Program A(i):
1: n:= 0
2: Print n mod 2
3: n:= n+1
4: If n=i Goto 6
5: Goto 2
6: End

The length of A (in bits) is an upper bound of K(010101...01).

Behavioural classes of abstract computing machines

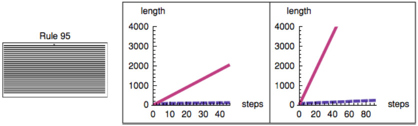
Wolfram's computational classes of behaviour:

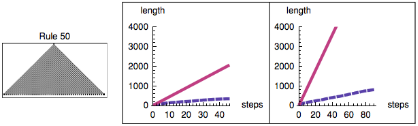
- Class I: system evolves into a homogeneous state.
- Class II: system evolves in a periodic state.
- Class III: system evolves into random-looking states.
- Class IV: system evolves into localised *complex structures.*

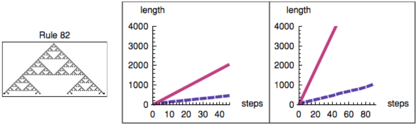
Living systems clearly fall in class IV systems.

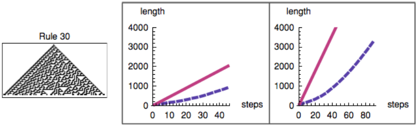
[Wolfram, (1994)]

Asymptotic behaviour of compressed evolutions





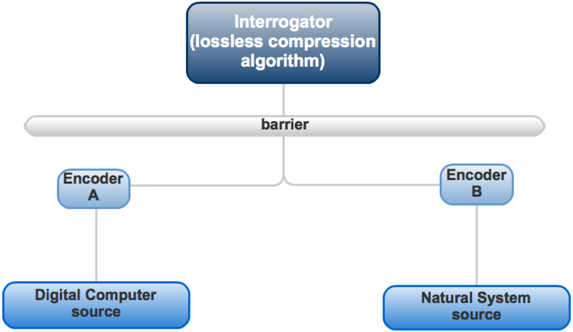




[Zenil, Complex Systems (2010)]

A behavioural approach to computation (and life recognition)

A Turing-test like test strategy to the question of life (instead of Turing's original question of artificial intelligence):



[Zenil, Philosophy & Technology and SAPERE, (2013)]

Answering Chalmer's rock question

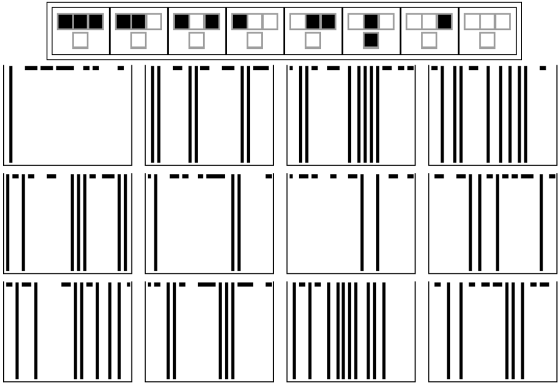


Figure : ECA R4 has a low C_n^t value for any n and t (it doesn't react *much to external stimuli).*

[Zenil, Philosophy & Technology, (2013)]

Turing universality

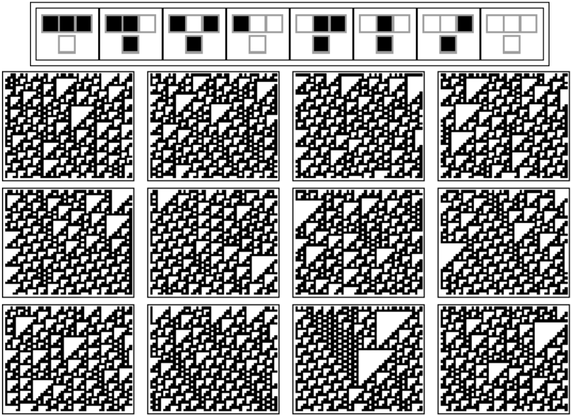
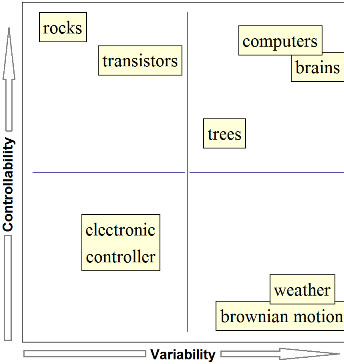


Figure : ECA R110 has large asymptotic coefficient C_n^t value for large enough choices of t and n, which is compatible with the fact that it is Turing universal (for particular semi-periodic initial configurations).

A hierarchical view of computing systems (and life?)

The measure can be applied to other than computers.
Programmability of physical and biological entities sorted by variability versus controllability:

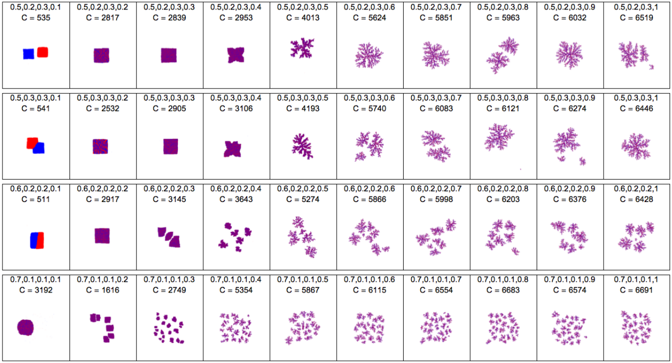


The diagonal determines the degree of programmability.

[Zenil, Ball, Tegnér, ECAL MIT Press Proceedings, (2013)]

Phorphyrins: mapping behaviour to stimuli

One can find the specific stimuli producing specific behaviours and go to the lab and test it. Compound parameters that have highest or lowest behavioural impact can be tested *and programmed in silico.*

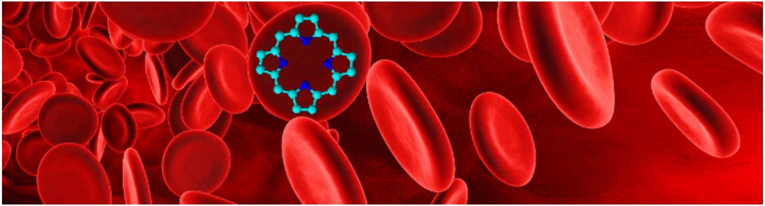


(Hemes in hemoglobin. They give the red color to blood)

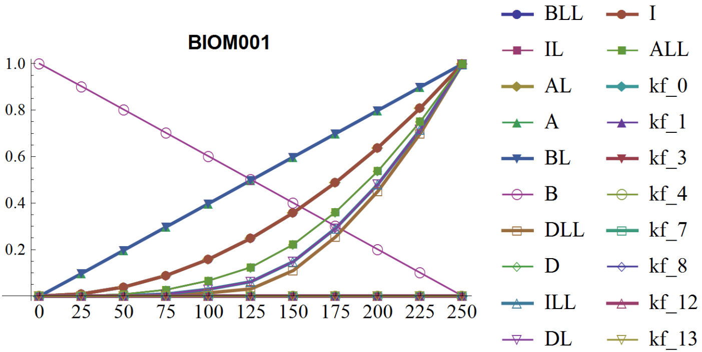
[Terrazas, Zenil and Krasnogor, Natural Computing (in press)]

Programmable Medicine

Building on molecular (DNA and non-DNA, e.g. porphyrin computation) imagine the introduction of an array of these molecules with a simple logic to treatments releasing a payload if expressing cancer markers, etc.

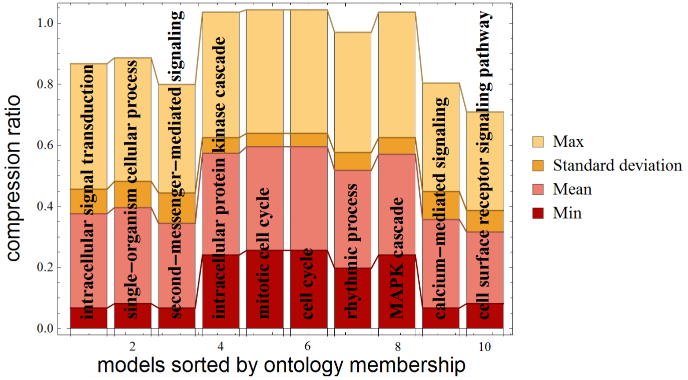


The EMBL-EBI BioModel's Database



The EMBL-EBI BioModel DB is a curated database of mathematical published (about 400) models related to living organisms.

Cell cycle versus metabolic pathways

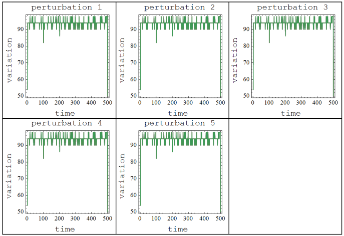


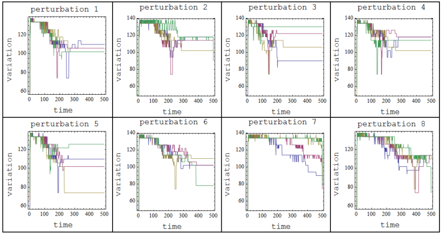
Variability analysis: Models cluster by their role in living processes.

[Zenil, Ball, Tegnér, ECAL MIT Press Proceedings, (2013)]

Variability and controllability of biological models

Information content trajectories
(from EMBL-EBI model simulations):





Programmability is a combination of variability and controllability

[Zenil, Ball, Tegnér, ECAL MIT Press Proceedings, (2013)]

- H. Zenil, Compression-based Investigation of the Dynamical Properties of Cellular Automata and Other Systems, ***Complex Systems,*** Vol. 19, No. 1, pages 1-28, 2010.
- H. Zenil, What is Nature-like Computation? A Behavioural Approach and a Notion of Programmability, ***Philosophy & Technology*** (special issue on History and Philosophy of Computing), 2013.
- H. Zenil, On the Dynamic Qualitative Behavior of Universal Computation ***Complex Systems,*** vol. 20, No. 3, pp. 265-278, 2012.
- G. Terrazas, H. Zenil and N. Krasnogor, Exploring Programmable Self-Assembly in Non DNA-based Computing, ***Natural Computing***, DOI: 10.1007/s11047-013-9397-2.
- H. Zenil and E. Villarreal-Zapata, Asymptotic Behaviour and Ratios of Complexity in Cellular Automata Rule Spaces, ***Journal of Bifurcation and Chaos*** (in press).
- H. Zenil, G. Ball and J. Tegnér, Testing Biological Models for Non-linear Sensitivity with a Programmability Test. In P. Liò, O. Miglino, G. Nicosia, S. Nolfi and M. Pavone (eds), ***Advances in Artificial Intelligence***, ECAL 2013, pp. 1222-1223, MIT Press, 2013.
- H. Zenil, A Turing Test-Inspired Approach to Natural Computation. In G. Primiero and L. De Mol (eds.), ***Turing in Context II (Brussels, 10-12 October 2012),***

Historical and Contemporary Research in Logic, Computing Machinery and Artificial Intelligence, Proceedings published by the Royal Flemish Academy of Belgium for Science and Arts, 2013.

- A Behavioural Foundation for Natural Computing and a Programmability Test. In G. Dodig-Crnkovic and R. Giovagnoli (eds), *Computing Nature: Turing Centenary Perspective*, SAPERE Series vol. 7, Springer, 2013.
- H. Zenil, Turing Patterns with Turing Machines: Emergence and Low-level Structure Formation, *Natural Computing,* 12(2): 291-303 (2013), 2013.
- J.-P. Delahaye and H. Zenil, Numerical Evaluation of the Complexity of Short Strings: A Glance Into the Innermost Structure of Algorithmic Randomness, *Applied Mathematics and Computation* 219, pp. 63-77, 2012.