In Pre-Proceedings of the 5th Int. Workshop on Information Processing in Cells and Tissues (IPCAT03), pp.257-274.
Swiss Federal Institute of Technology, Lausanne, Switzerland.

Artificial Cell Division

Daniel Mange, André Stauffer, Enrico Petraglio,
and Gianluca Tempesti

Swiss Federal Institute of Technology
Logic Systems Laboratory
CH-1015 Lausanne, Switzerland
daniel.mange@epfl.ch

Abstract

After a survey of the theory and some realizations of self-replicating machines, this paper presents a novel self-replicating loop endowed with universal construction and computation properties. Based on the hardware implementation of the so-called Tom Thumb algorithm, the design of this loop leads to a new kind of cellular automaton made of a processing and a control units. The self-replication of the Swiss flag serves as an artificial cell division example of the loop which, according to autopoietic evaluation criteria, corresponds to a cell showing the phenomenology of a living system.

Key words: self-replication, universal construction, universal computation, cellular automaton, artificial cell division, autopoiesis

1 Introduction and survey

1.1 John von Neumann's self-replicating automaton

The main goal of this paper is to present a new self-replicating machine endowed with universal construction and computation properties. The early history of the theory of self-replicating machines is basically the history of John von Neumann's thinking on the matter [20]. Von Neumann's cellular automaton, as well as all the machines described in this paper, is based on the following general hypotheses.

- The automaton deals exclusively with the flow of information; the physical material (usually a silicon substrate) and the energy (power supply) are given a priori.

- The physical space is two-dimensional and as large as desired.
- The physical space is *homogeneous*, that is comprised by identical *molecules*[1], all of which have the same internal architecture and the same connections with their neighbors; only the *state* of a molecule (the combination of the values in its memories) can distinguish it from its neighbors.
- Replication is considered as a special case of growth: this process involves the creation of an identical organism by duplicating the genetic material of a mother entity onto a daughter one, thereby creating an exact clone.

In his historical work, von Neumann showed that a possible *configuration* (a set of molecules in a given state) of his automaton can implement a universal constructor (Uconst) endowed of the three following properties.

(1) *Universal construction*: given the one-dimensional description of any two-dimensional machine M (i.e. a machine of any size), the universal constructor can build an exact copy M' of this machine in the molecular space.

(2) *Self-replication*: given the description of the constructor itself, it is then possible to build a copy of the constructor in the molecular space; the constructor first interprets the description D(Uconst) to build a copy Uconst' whose memory is empty (*translation process*), and then copies the description D(Uconst) from the original memory of Uconst to the memory of Uconst' (*transcription process*).

(3) *Universal computation*: by attaching to the constructor a universal computer (a universal Turing machine), and by placing the description of both the constructor and the universal computer in the original memory, the universal constructor produces a copy of itself and a copy of the universal computer through the mechanism described above.

According to the biological definition of a cell, Von Neumann's automaton is a unicellular organism: its genome is composed of the description of the machine M to be constructed written in the memory of the constructor.

The dimensions of von Neumann's automaton are substantial (in the order of 200,000 molecules); it has thus never been physically implemented and has been simulated only partially [12]. If von Neumann and his successors Burks, Thatcher, Lee, Codd, Banks, Nourai and Kashef demonstrated the theoretical possibility of realizing self-replicating automata with universal calculation [9], a practical implementation requires a sharply different approach. It was finally Langton, in 1984, who opened a second stage in this field of research.

[1] To avoid conflicts with biological definitions, we do not use the term "cell" to indicate the parts of a cellular automaton, opting rather for the term "molecule". In fact, in biological terms, a *cell* can be defined as the smallest part of a living being which carries the complete blueprint of the being, that is the being's *genome*.

1.2 Self-replicating loops

In order to construct a self-replicating automaton simpler than that of von Neumann, Langton [6] adopted more liberal criteria: he dropped the condition that the self-replicating unit must be capable of universal construction and computation.

Langton proposes a configuration in the form of a loop, endowed notably of a constructing arm and of a replication program or genome, which turns counterclockwise. After 151 time steps, the original loop (mother loop) produces a daughter loop, thus obtaining the self-replication of Langton's loop.

There is no universal construction nor calculation: the loop does nothing but replicate itself. Langton's self-replicating loop represents therefore a special case of von Neumann's self-replication of a universal constructor. The loop is a non-universal constructor, capable of building, on the basis of its genome, a single type of machine: itself.

Referring to biological definitions again, Langton's self-replicating loop is a unicellular organism: its genome requires 28 molecules and is a subset of the complete loop which requires 94 molecules.

As did von Neumann, Langton emphasized the two different modes in which information is used, interpreted (*translation*) and uninterpreted (*transcription*). In his loop, translation is accomplished when the instruction signals are executed as they reach the end of the construction arm, and upon collision of signals with other signals. Transcription is accomplished by duplication of signals at the arm junctions.

The size of Langton's loop is perfectly reasonable, since it requires 94 molecules, thus allowing complete simulation. More recently, Byl [1] proposed a simplified version of Langton's automaton. Last but not least Reggia *et al.* [13] discovered that having a sheath surrounding the data paths of the genome was not essential, and that its removal led to smaller self-replicating structures which also have simpler transitions functions.

1.3 Self-replicating loops with computing capabilities

All the previous loops lack any computing and constructing capabilities, their sole functionality being that of self-replication. Lately, new attemps have been made to redesign Langton's loop in order to embed such calculation possibilities. Tempesti's loop [17] is thus a self-replicating automaton, with an attached executable program that is duplicated and executed in each of the copies. This was demonstrated for a simple program that writes out (after the loop's replication) "LSL", acronym of the Logic Systems Laboratory. Finally, Perrier *et al.*'s self-replicating loop [11] shows some kind of universal computational capabilities. The system consists of three parts, loop, program, and data, all of which are replicated, followed by the program's execution on the given data.

So far, all self-replicating loops are lacking universal construction, i.e. the capability of constructing a two-dimensional computing machine of any dimensions, even if this goal is of highest interest for developing new cellular automata, for example the three-dimensional reversible cellular automata designed by Imai *et al.* [5] for the emerging field of nanotechnologies.

1.4 Self-replicating loops with universal construction and computation

Our goal is to show that a new algorithm, the *Tom Thumb algorithm*, will make it possible to design a self-replicating loop with universal construction and universal computation that can easily be implemented into silicon.

In Section 2, our new algorithm will be described by means of a minimal mother cell composed of four molecules which will grow and then divide, triggering the growth of two daughter cells. This example is sufficient for deriving the schematic architecture of the basic molecule. Section 3 deals with the generalization of the methodology previously described and its application to a real example, the self-replication of the Swiss flag. Universal construction and computation are briefly demonstrated. Section 4 will conclude by opening new avenues based on the self-replicating loop with universal construction.

2 A new algorithm for the artificial cell division

2.1 Cell division in living organisms

Before describing our new algorithm for the division of an artificial cell, let us remember the roles that cellular division plays in the existence of living organisms [2](p. 206).

"When a unicellular organism divides to form duplicate offspring, the division of a cell reproduces an entire organism. But cell division also enables multicellular organisms, including humans, to grow and develop from a single cell, the fertilized egg. Even after the organism is fully grown, cell division continues to function in renewal and repair, replacing cells that die from normal wear and tear or accidents. For example, dividing cells in your bone marrow continuously supply new blood cells. The reproduction of an ensemble as complex as a cell cannot occur by mere pinching in half; the cell is not like a soap bubble that simply enlarges and splits in two. Cell division involves the distribution of identical genetic material (DNA) to two daughter cells. What is most remarkable about cell division is the fidelity with which the DNA is passed along, without dilution, from one generation of cells to the next. A dividing cell duplicates its DNA, allocates the two copies to opposite ends of the cell, and only then splits into two daughter cells".

In conclusion, we can summarize the two key roles of cell division.

- The construction of two daughter cells in order to grow a new organism or to repair an already existing one (genome *translation*).
- The distribution of an identical set of chromosomes in order to create a copy of the genome from the mother cell aimed at programming the daughter cells (genome *transcription*).

Starting with a minimal cell made up of four artificial molecules, we will propose a new algorithm, the *Tom Thumb algorithm*, aimed at constructing both the daughter cells and the associated genomes. This algorithm will finally allow us to derive the schematic architecture of our final molecule. A tissue of such molecules will in the end be endowed of both universal construction and computation properties.

2.2 Initial conditions

The minimal cell compatible with our algorithm is made up of four molecules, organized as a square of two rows by two columns (Figure 1). Each molecule is able to store in its three memory positions three hexadecimal characters of our artificial genome, and the whole cell thus embeds twelve such characters.

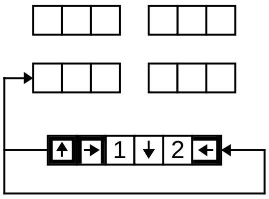
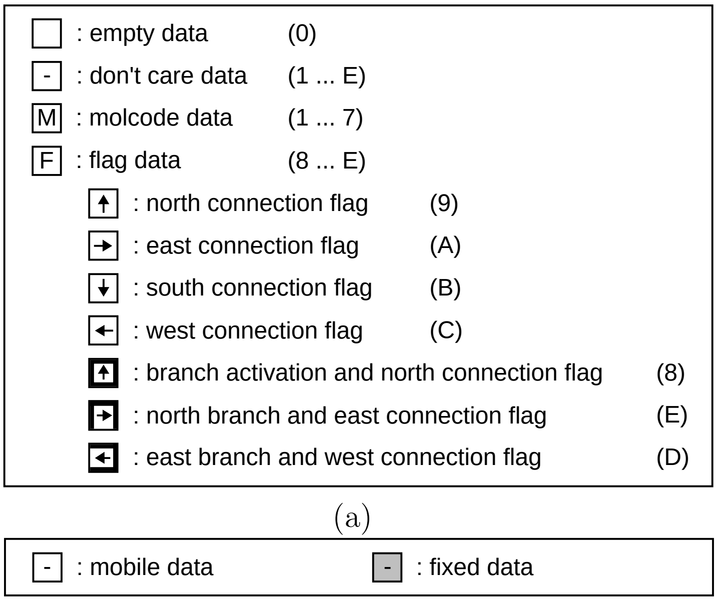


Fig. 1. The minimal cell (2 × 2 molecules) with its genome at the start ($t = 0$).

The original genome for the minimal cell is organized as a string of six hexadecimal characters, i.e. half the number of characters in the cell, moving counterclockwise by one character at each time step ($t = 0, 1, 2, ...$).

The 15 used hexadecimal characters composing the alphabet of our artificial genome are detailed in Figure 2. They are either *empty data* (0), *molcode data* (for molecule code data, from 1 to 7) or *flag data* (from 8 to E). Molcode data will be used for configuring our final artificial organism, while flag data are indispensable for constructing the skeleton of the cell. Furthermore, each character is given a status and will eventually be *mobile data*, indefinitely moving around the cell, or *fixed data*, definitely trapped in a memory position of the cell.


: empty data (0)
- : don't care data (1 ... E)
M : molcode data (1 ... 7)
F : flag data (8 ... E)
: north connection flag (9)
: east connection flag (A)
: south connection flag (B)
: west connection flag (C)
: branch activation and north connection flag (8)
: north branch and east connection flag (E)
: east branch and west connection flag (D)
- : mobile data
- : fixed data

(a)

(b)

Fig. 2. The 15 characters forming the alphabet of an artificial genome. (a) Graphical and hexadecimal representations of the 15 characters. (b) Graphical representation of the status of each character.

2.3 Constructing the cell

At each time step, a character of the original genome is shifted from right to left and simultaneously stored in the lower leftmost molecule (Figures 1 and 3). The construction of the cell, i.e. storing the fixed data and defining the paths for mobile data, depends on three patterns (Figure 4).

- If the two rightmost memory positions of a molecule are empty (blank squares), the characters are shifted by one position to the right (shift data). Remark that the first character is always a flag *F* due to our algorithm.
- If the rightmost memory position is empty and the two leftmost memory positions hold flags (*F*), the characters are shifted by one position to the right (load flag). In this situation, the rightmost *F* character is trapped in the molecule (fixed data), and a new connection is established from the central position toward the northern, eastern, southern or western molecule,

depending on the fixed flag information (F = 9, A, B or C).

- If the rightmost memory position is empty, while the central and leftmost memory positions hold a flag (F) and a molcode (M) respectively, then the characters are shifted by one position to the right (load molcode and flag). In this case, both characters are trapped in the molecule (fixed data), and a new connection is launched from the leftmost position toward the northern, eastern, southern or western molecule, depending on the fixed flag information (F = 9, A, B or C).

At time $t = 12$, twelve characters, i.e. twice the contents of the original genome, have been stored in the twelve memory positions of the cell (Figure 3). Six characters are fixed data, forming the skeleton of the final cell, and the six remaining ones are mobile data, composing a copy of the original genome. Both *translation* (i.e. construction of the cell) and *transcription* (i.e. copy of the genetic information) have been therefore achieved.

The fixed data trapped in the rightmost memory position(s) of each molecule remind us of the pebbles left by Tom Thumb for memorizing his way.

2.4 Dividing the mother cell into two daughter cells

In order to grow an artificial organism in both horizontal and vertical directions, the mother cell should be able to trigger the construction of two daughter cells, nothward and eastward.

At time $t = 8$ (Figure 3), we observe a pattern of characters which is able to start the construction of the northward daughter cell; the upper leftmost molecule is characterized by two specific signals, i.e. a fixed flag indicating a north branch (F = E) and a branch activation flag (F = 8) ready to enter the leftmost memory position. This pattern is also visible in Figure 5 (northward signal, third row).

At time $t = 17$, another particular pattern of characters will start the construction of the eastward daughter cell; the lower rightmost molecule is characterized by two specific signals, i.e. a fixed flag indicating an east branch (F = D), and the branch activation flag (F = 8) in the leftmost memory position. This pattern appears also in Figure 5 (eastward signal, third row).

The other patterns in Figure 5 are needed for constructing the inner paths of a cell more complex than the minimal cell, for example that of Figure 8b.

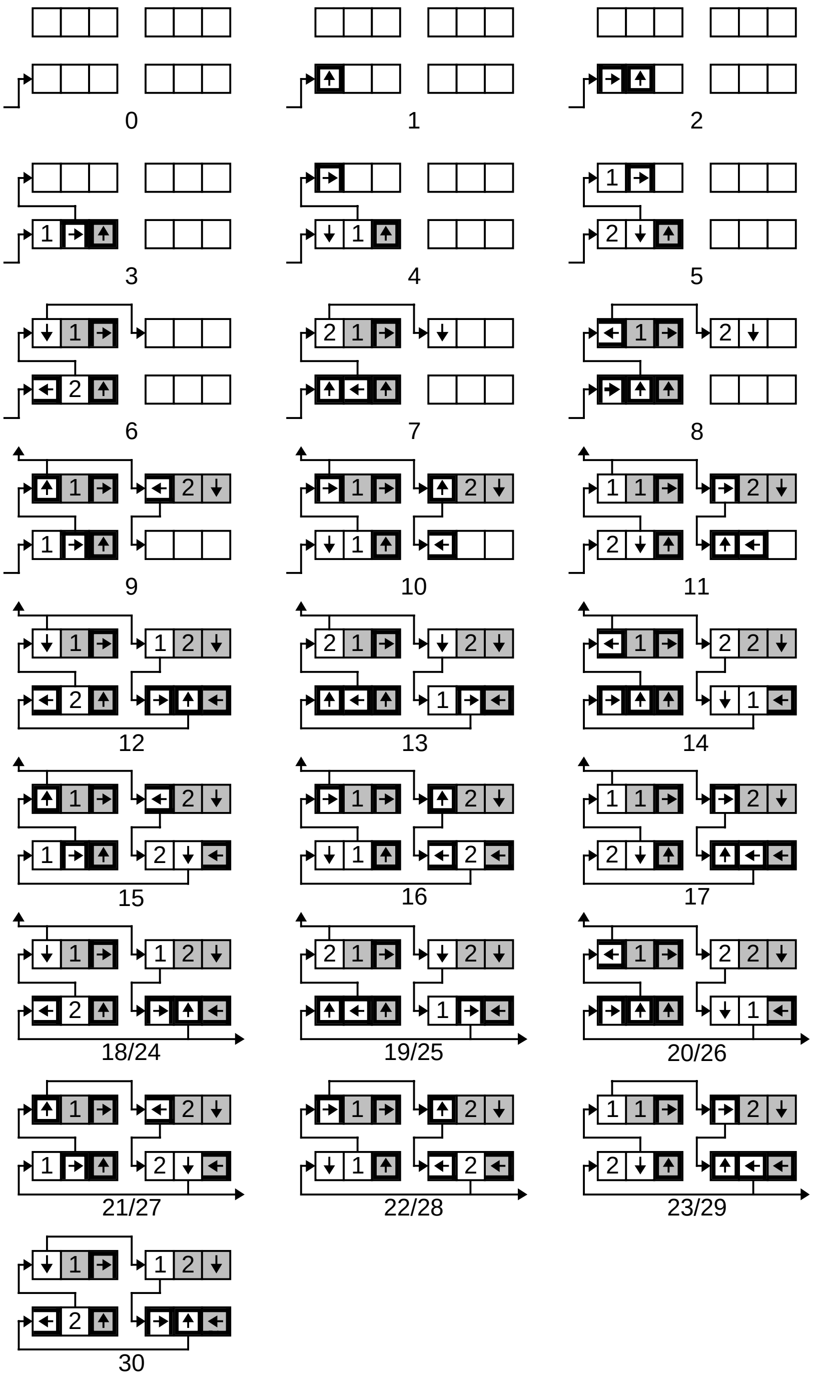


Fig. 3. Constructing the minimal cell.

2.5 Growing a multicellular organism

In order to analyze the growth of a multicellular artificial organism, we are led to carefully observe the interactions of the different paths created inside and outside each individual cell. This analysis, which is beyond the scope of this introductory paper, will be detailed elsewhere.

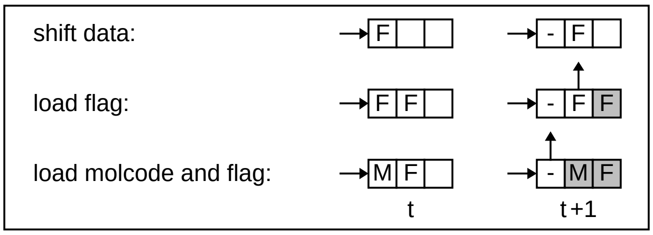


Fig. 4. The three memory patterns for constructing a cell.

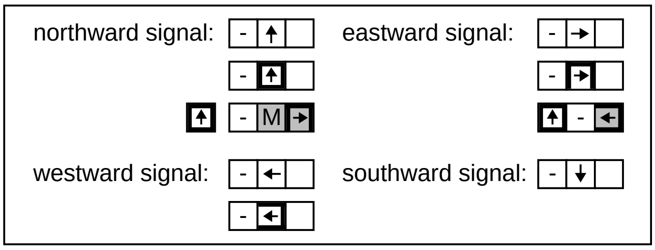


Fig. 5. Patterns of characters triggering the paths to the north, east, south and west molecules.

We finally made the following choice: a closing loop has priority over all other outer paths, which makes the completed loop entirely independent of its neighbors, and the organism will grow by developing bottom-up vertical branches. This choice is quite arbitrary and may be changed according to other specifications.

It is now possible to come back to the detailed representation of a multicellular organism made up of 2×2 minimal cells (Figure 6) and exhibit it at different time steps in accordance with the above mentioned priorities.

2.6 Toward a hardware implementation

We are now able to describe the schematic architecture of our actual molecule (Figure 7) which is made up of two main parts, a *processing unit* and a *control unit*. The *processing unit* is itself decomposed into three units.

- An input unit, the multiplexer DIMUX, selecting one out of the four input data ($NDI3{:}0$, $EDI3{:}0$, $SDI3{:}0$ or $WDI3{:}0$) plus the empty data 0000; this selection is operated by a 3-bit control signal $I2{:}0$.
- A 3-level stack organized as a propagation register P3:0 (for mobile data), a molcode register M3:0 (for mobile data or fixed molcode), and a flag register F3:0 (for fixed flag) according to the definitions of Figure 4.
- An output unit, the multiplexer DOMUX, selecting either the propagation register or the molcode register; this selection is performed by a 2-bit control signal $O1{:}0$, itself depending on the $P3$, $M3$ and $F3$ variables according to the rules described in Figure 4.

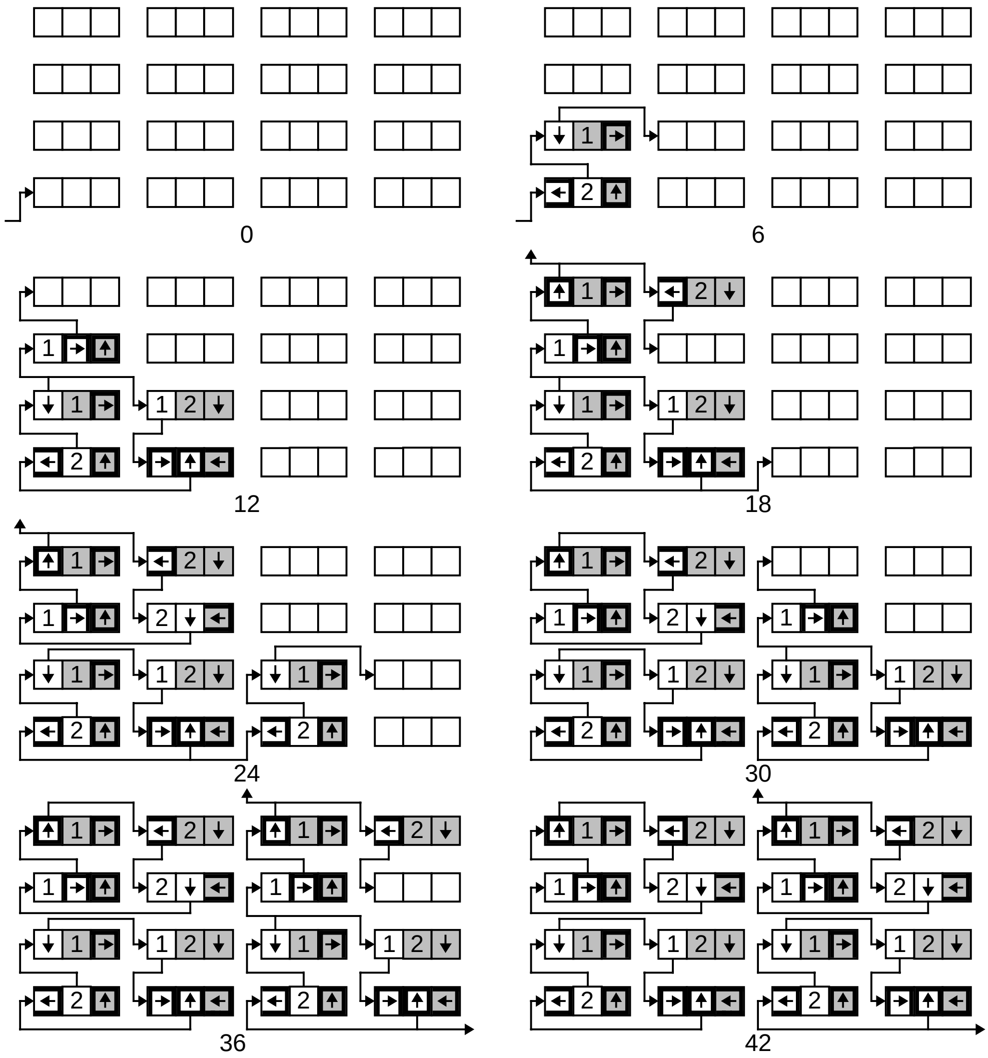


Fig. 6. Analyzing a multicellular organism made up of 2 × 2 minimal cells.

The *control unit* is itself decomposed into two units.

- An input encoder ENC, a finite state machine calculating the 3-bit control signal $I2{:}0$ from the four input signals NSI, ESI, SSI, and WSI. The specification of this machine, which depends on the priorities between cells as mentioned above, is beyond the scope of this introductory paper and will be detailed elsewhere.
- An output generator GEN, which is a combinational system producing the northward, eastward, southward, and westward signals (NSO, ESO, SSO, and WSO) according to the patterns described in Figure 5.

A look at Figure 7 allows the calculation of the number of the state variables involved in the molecule, i.e. twelve for the stack ($P3{:}0$, $M3{:}0$, and $F3{:}0$), three for the control signals of the input multiplexer ($I2{:}0$), and two for the control signals of the output multiplexer ($O1{:}0$), which amount to a total of 17.

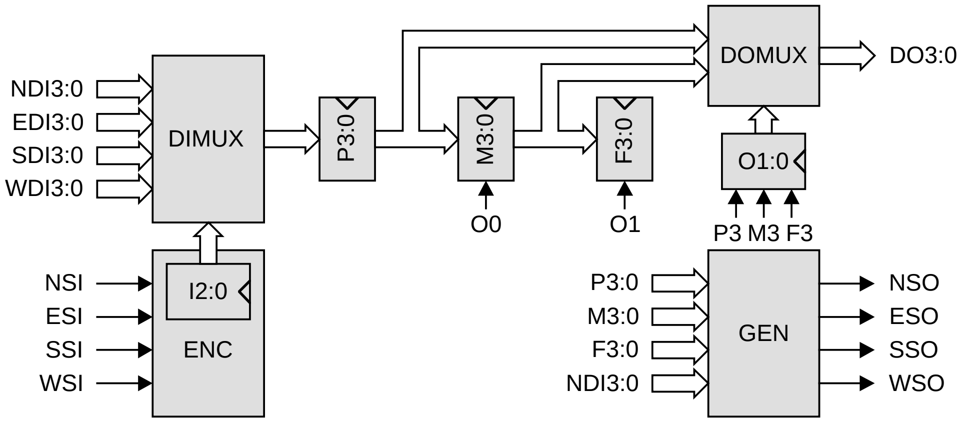


Fig. 7. Schematic architecture of the basic molecule.

Therefore, the number of possible states is 2^{17}. Thanks to our methodology, i.e. decomposing the molecule into a processing unit and a control unit, we have no need to carry out the whole state table with 2^{17} rules and we are able to synthesize our final architecture in a straightforward way.

Moreover, we will show in the next Section that both the software (i.e. our artificial genome) and the hardware (i.e. the architecture) of our molecule are completely scalable, i.e. may be adapted to any given application.

3 Design methodology and generalization

3.1 A design example

In [17], Tempesti has already shown how to embed the acronym "LSL" (for Logic Systems Laboratory) into a self-replicating loop implemented on a classical cellular automaton. Thanks to a "cut-and-try" methodology and a powerful software tool, he was able to carry out the painful derivation of over ten thousand rules for the basic cell.

Unlike the heuristic method used by Tempesti, we will show that an example of comparable complexity, the Swiss flag, can be designed in a straightforward and systematic way thanks to the use of our new cellular automaton associated to the Tom Thumb algorithm.

The Swiss flag is first represented in a rectangular array of 8 columns by 7 rows (Figure 8a). While the number of rows is indifferent, the number of columns should be even in order to properly close the loop (Figure 8b). The cell is therefore made up of 8 × 7 = 56 molecules connected according to the pattern in Figure 8b: bottom-up in the odd columns, top-down in the even columns, with the lower row reserved for closing the loop. It is then possible to define all the flags in the rightmost memory position of each molecule (grey characters in Figure 8b) without forgetting the branch activation and north connection flag in the lower molecule of the first column, the north branch and east connection flag in the upper molecule of the first column, and the east branch and west connection flag in the lower molecule of the last column.

According to our algorithm, half of the 56 molecules (28) are *phenotypic molecules*, i.e. storing a molcode for displaying the Swiss flag, the other ones being *genotypic molecules* necessary for circulating a complete copy of the genome. Among the 28 phenotypic molecules, 20 are used for displaying the background of the flag, and are given the character “2” as molcode (black data in Figures 8a and 8b), while 5 are used for displaying the cross (molcode “1”) and the last 3 are blank (molcode “0”). We have chosen quite arbitrarily to distribute these latter ones in three corners of the cell.

The other 28 genotypic molecules are kept for circulating the final genome whose detailed information, i.e. $28 \times 3 = 84$ hexadecimal characters (Figure 8c), is derived by reading clockwise the fixed characters (black and grey characters in Figure 8b) of the whole loop, starting with the lower molecule of the first column. Finally, we just assume that each genotypic molecule will produce a blank display in order to respect the original specifications.

Last, it was possible to embed the basic molecule of Figure 7 in each of the 2000 field-programmable gate arrays of the BioWall [18] and to show the rather spectacular self-replication of our original cell (equivalent to a unicellular artificial organism), the Swiss flag, in both the vertical and horizontal directions (Figure 8d).

3.2 Software and hardware scalability

Assuming the existence of a two-dimensional array of molecules as implemented in Figure 7, it is possible to configure any artificial organism characterized by one molcode (i.e. one hexadecimal character) for each molecule. The molcode may be directly used, as in the previous example, for displaying the given specifications or may configure any kind of field-programmable gate array aimed at defining a more complex digital architecture. There are only two restrictions involved by our algorithm.

(1) An even number of rows and/or columns, in order to properly close the loop.
(2) A sufficient number of molecules for embedding both the artificial organism, i.e. the phenotype, and its description, i.e. its genome: if $M/2$ is the number of molecules necessary for the construction of the artificial

organism, we need in the worst case $M/2$ molecules for its description, i.e. a total of M molecules.

(3) Constraints on the upper leftmost and lower rightmost molecules of the cell allowing the construction of the outer paths: they should be phenotypic respectively genotypic molecules.

If these three conditions are met, the software scalability of our loop is guaranteed.

For any artificial organism characterized by more than one molcode (i.e. more than one hexadecimal character) in each phenotypic molecule, we are led to slightly modify the architecture of Figure 7 and to introduce as many molcode registers M3:0 in the stack. As required, this modification will guarantee the hardware scalability of the loop.

As both software and hardware scalabilities are verified, we may claim that our loop guarantees *universal construction*, i.e. the construction of digital systems of any dimensions in both the horizontal and vertical directions. We have therefore proved that our loop is really endowed of universal construction.

On the other hand, we have already shown that a universal Turing machine may be embedded in a regular array of identical cells [14], themselves decomposed and implemented onto a regular array of molecules. Our new loop with universal construction can therefore verify *universal computation*, thus meeting the two basic properties of the historical self-replicating cellular automaton designed by von Neumann [20], i.e. *universal construction and computation.*

4 Conclusion

4.1 Present and future applications

Several years before the publication of the historical paper by Crick and Watson [21] revealing the existence and the detailed architecture of the DNA double helix, von Neumann was already able to point out that a self-replicating machine required the existence of a one-dimensional description, the genome, and a universal constructor able to both interpret (translation process) and copy (transcription process) the genome in order to produce a valid daughter organism. Self-replication allows not only to divide a mother cell (artificial or living) into two daughter cells, but also to grow and repair a complete organism. Self-replication is now considered as a central mechanism indispensable for those circuits which will be implemented through the nascent field of nanotechnologies [15] [4].

A first field of application of our new self-replicating loop with universal construction is quite naturally the classical self-replicating automata, such as three-dimensional reversible automata [5] or asynchronous cellular automata [10].

A second, and possibly more important field of application is Embryonics, where artificial multicellular organisms are based on the growth of a cluster of cells, themselves produced by cellular division [7] [8].

A major by-product of this research is the introduction of a new kind of cellular automaton, decomposed in a processing and a control units, which allows for a systematic and straightforward design methodology which is lacking at the moment.

Other possible open avenues concern the evolution of such loops and/or their capability to carry out massive parallel computation [3].

4.2 Is our loop an autopoietic machine?

In an historical paper published in 1974 [19], Varela, Maturana and Uribe proposed a concise set of criteria for determining whether or not a "machine" is an "autopoietic machine". The criteria are presented in the form of a 6-point checklist by which one may proceed step-by-step in evaluating autopoiesis for a given unity.

(1) "Determine if the unity has identifiable boundaries". Our cell has clearly identifiable boundaries defined by the fixed flags of its corner molecules and by the connections between these molecules.
(2) "Determine if there are constitutive elements of the unity, that is, components of the unity". Our cell is made up of a set of parts, its molecules.
(3) "Determine if the unity is a mechanistic system, that is, the component properties are capable of satisfying certain relations that determine in the unity the interactions and transformations of these components". In the simple example of Figure 8b, the set of molecules is necessary for displaying the desired behavior, i.e. the swiss flag. In the more general case, remember that the molcodes are configuring field-programmable gate arrays whose interactions may be very complex, as already shown in the Embryonics project [16].
(4) "Determine if the components that constitute the boundaries of the unity constitute these boundaries through preferential neighborhood relations and interactions between themselves, as determined by their properties in the space of their interactions". The molecules which constitute the boundaries of our cell are characterized by specific fixed flags and connections which are the direct result of their mutual interactions.

(5) "Determine if the components of the boundaries of the unity are produced by the interactions of the components of the unity, either by transformation of previously produced components, or by transformations and/or coupling of non-component elements that enter the unity through its boundaries". The molecules of the apparent boundary are produced by the process constructing the cell itself, i.e. the Tom Thumb algorithm which transforms empty molecules into active molecules thanks to non-component elements that enter the cell through its boundaries, i.e. the artificial genome.

(6) "Determine if all other components of the unity are also produced by interactions of its components as in 5, and if those which are not produced by the interactions of other components participate as necessary permanent constitutive components in the production of other components". All the molecules of the cell are progressively constructed by the other molecules within the cell itself thanks to non-component elements, i.e. the artificial genome.

Our self-replicating loop with universal construction is thus an autopoietic cell. According to Varela *et al.*, such an autopoietic cell has the phenomenology of a living system.

Acknowledgments

This work was supported in part by the Swiss National Science Foundation under grant 20-100049.1, by the Leenaards Foundation, Lausanne, Switzerland, and by the Villa Reuge, Ste-Croix, Switzerland.

We would like to thank the reviewers for their invaluable contribution.

We also thank Nicolas Mange for his contribution.

References

[1] J. Byl. Self-reproduction in small cellular automata. *Physica D*, 34:295–299, 1989.

[2] N. A. Campbell, J. B. Reece, and L. G. Mitchell. *Biology, 5th edition.* Benjamin/Cummings, Menlo Park, 1999.

[3] H.-H. Chou and J. A. Reggia. Problem solving during artificial selection of self-replicating loops. *Physica D*, 115(3-4):293–312, 1998.

[4] K. E. Drexler. *Nanosystems: Molecular Machinery, Manufacturing, and Computation.* John Wiley, New York, 1992.

[5] K. Imai, T. Hori, and K. Morita. Self-reproduction in three-dimensional reversible cellular space. *Artificial Life*, 8(2):155–174, 2002.

[6] C. G. Langton. Self-reproduction in cellular automata. *Physica D*, 10:135–144, 1984.

[7] N. J. Macias and L. J. K. Durbeck. Self-assembling circuits with autonomous fault handling. In A. Stoica, J. Lohn, R. Katz, D. Keymeulen, and R. S. Zebulum, editors, *Proceedings of the 2002 NASA/DOD Workshop Conference on Evolvable Hardware*, pages 46–55, Los Alamitos, CA, 2002. IEEE Computer Society Press.

[8] D. Mange, M. Sipper, A. Stauffer, and G. Tempesti. Toward robust integrated circuits: The Embryonics approach. *Proceedings of the IEEE*, 88(4):516–541, April 2000.

[9] D. Mange and M. Tomassini, editors. *Bio-Inspired Computing Machines*. Presses polytechniques et universitaires romandes, Lausanne, 1998.

[10] C. L. Nehaniv. Self-reproduction in asynchronous cellular automaton. In A. Stoica, J. Lohn, R. Katz, D. Keymeulen, and R. S. Zebulum, editors, *Proceedings of the 2002 NASA/DOD Workshop Conference on Evolvable Hardware*, pages 201–209, Los Alamitos, CA, 2002. IEEE Computer Society Press.

[11] J.-Y. Perrier, M. Sipper, and J. Zahnd. Toward a viable, self-reproducing universal computer. *Physica D*, 97:335–352, 1996.

[12] U. Pesavento. An implementation of von Neumann's self-reproducing machine. *Artificial Life*, 2(4):337–354, 1995.

[13] J. A. Reggia, S. L. Armentrout, H.-H. Chou, and Y. Peng. Simple systems that exhibit self-directed replication. *Science*, 259:1282–1287, February 1993.

[14] H. F. Restrepo and D. Mange. An embryonics implementation of a self-replicating universal Turing machine. In Y. Liu, K. Tanaka, M. Iwata, T. Higuchi, and M. Yasunaga, editors, *Evolvable Systems: From Biology to Hardware (ICES 2001)*, volume 2210 of *Lecture Notes in Computer Science*, pages 74–87. Springer-Verlag, Berlin, 2001.

[15] M. C. Roco and W. S. Bainbridge, editors. *Converging technologies for improving human performance. Nanotechnology, biotechnology, information technology and cognitive science.* NSF/DOC - sponsored report, Arlington, VA, 2002.

[16] A. Stauffer, D. Mange, G. Tempesti, and C. Teuscher. Biowatch: A giant electronic bio-inspired watch. In D. Keymeulen, A. Stoica, J. Lohn, and R. S. Zebulum, editors, *Proceedings of the Third NASA/DOD Workshop on Evolvable Hardware*, pages 185–192, Los Alamitos, CA, 2001. IEEE Computer Society.

[17] G. Tempesti. A new self-reproducing cellular automaton capable of construction and computation. In F. Morán, A. Moreno, J. J. Merelo, and P. Chacón, editors, *ECAL'95: Third European Conference on Artificial Life*, volume 929 of *Lecture Notes in Computer Science*, pages 555–563, Heidelberg, 1995. Springer-Verlag.

[18] G. Tempesti, D. Mange, A. Stauffer, and C. Teuscher. The BioWall: An electronic tissue for prototyping bio-inspired systems. In A. Stoica, J. Lohn, R. Katz, D. Keymeulen, and R. S. Zebulum, editors, *Proceedings of the 2002 NASA/DOD Workshop Conference on Evolvable Hardware*, pages 221–230, Los Alamitos, CA, 2002. IEEE Computer Society Press.

[19] F. G. Varela, H. R. Maturana, and R. Uribe. Autopoiesis: the organization of living systems, its characterization and a model. *Bio Systems*, 5:187–196, 1974.

[20] J. von Neumann. *Theory of Self-Reproducing Automata*. University of Illinois Press, Illinois, 1966. Edited and completed by A. W. Burks.

[21] J. D. Watson and F. H. C. Crick. A structure for desoxyribose nucleid acid. *Nature*, 171:737–738, 1953.

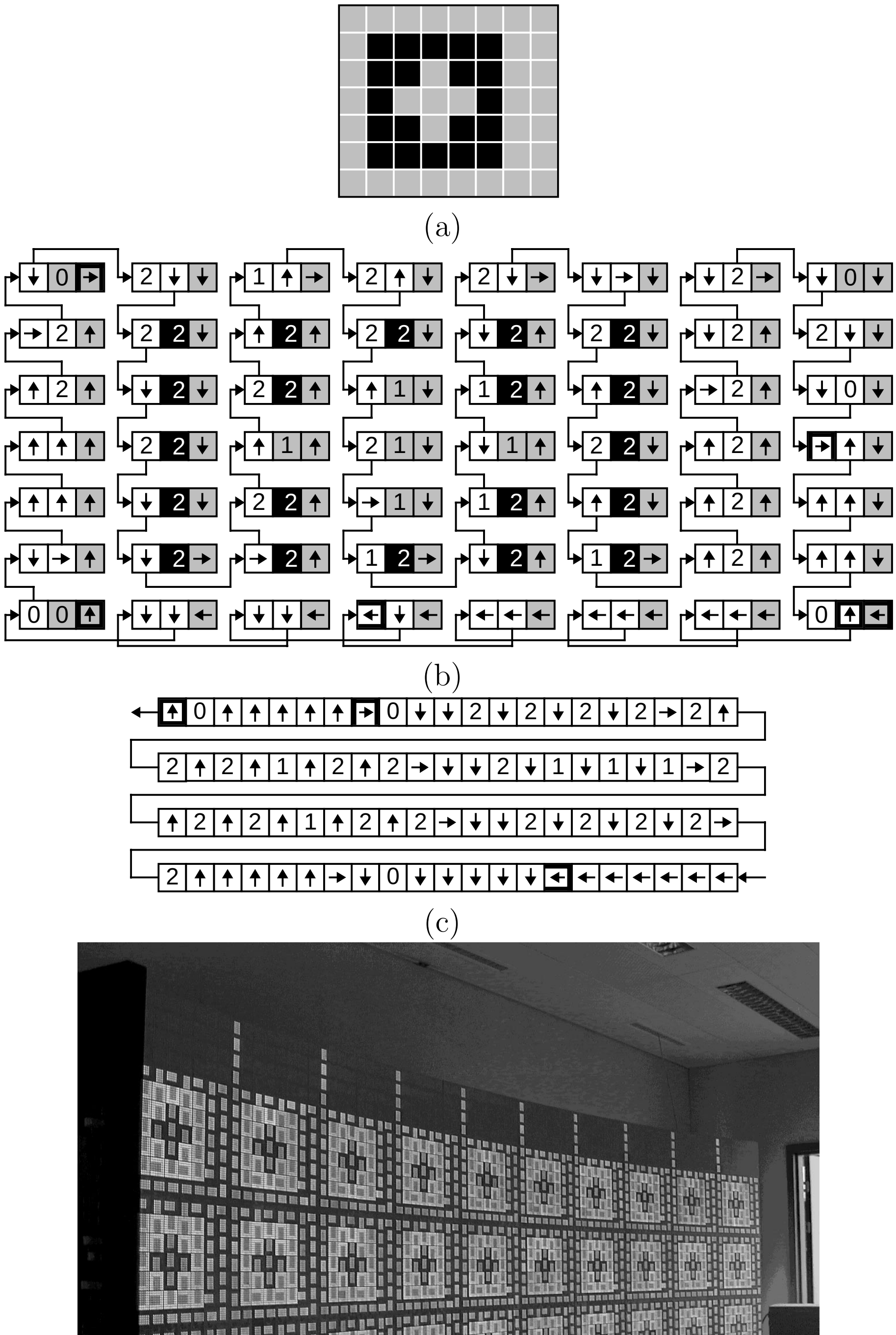


Fig. 8. Self-replication of the Swiss flag. (a) Original specifications. (b) The 8×7 = 56 molecules of the basic cell. (c) Genome. (d) BioWall implementation showing both the genotypic data path and the phenotypic flag (Photograph by A. Badertscher).