Tom Thumb Algorithm and von Neumann Universal Constructor

Joël Rossier, Enrico Petraglio, André Stauffer, and Gianluca Tempesti

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

Abstract. This article describes the addition to the von Neumann cellular automaton of the Tom Thumb Algorithm, a mechanism developed for the self-replication of multi-processor systems. Except for the cell construction process, every functionality of the original CA has been preserved in our new system. Moreover, the Tom Thumb Algorithm now allows the replication of any structure within the von Neumann environment, whatever its number of cells may be.

1 Introduction

In the middle of the last century, John von Neumann was interested in the concept of self-replication. During his research, he presented a model for an Universal Constructor (UC) system based on a two-dimensional cellular automaton of 29 states [1]. This UC is divided in two different parts: a tape that contains the description of the cellular machine to construct and the constructor in itself that reads the tape and builds the new corresponding machine with a dedicated cellular arm. Moreover, if the tape contains the description of the constructor and if the constructor, in addition to its building abilities, can also copy the tape information, then the resulting system is a self-replicating one (figure 1).

In spite of the conceptual success of von Neumann's work, the realization of such a system encounters a major problem: with the proposed architecture, the UC in itself needs about 200'000 cells, while the tape containing its description is almost five time greater. Clearly, it is too big for a physical implementation (at least today) and it is a real challenge even for a software simulation. In fact, the UC has not yet been completely specified [2].

In 1995, Umberto Pesavento presented an extension of the transition rules of the standard system [3]. Thus, he drastically reduced the number of cells used in the UC replication and proved the validity of the von Neumann self-replication process by simulating the entire UC. Despite that, once again, there were too many cells for a physical implementation. To solve this problem, we decided to apply to the standard von Neumann system a new kind of self-replicating process that uses a smaller number of cells: the Tom Thumb Algorithm (TTA).

In section 2, we will briefly present the standard von Neumann system. The basic operation of the Tom Thumb Algorithm will be exposed in section 3.

P.M.A. Sloot, B. Chopard, and A.G. Hoekstra (Eds.): ACRI 2004, LNCS 3305, pp. 1–10, 2004.

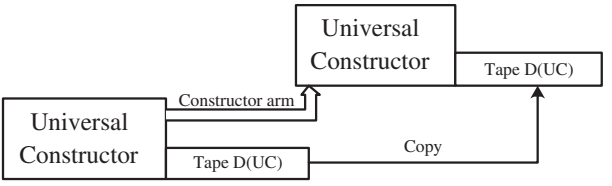

Fig. 1. Universal Constructor self-replication

Section 4 will then introduce the system we have developed for self-replication in the von Neumann environment and in section 5, we will discuss of some potentialities of such a realization from an evolutionary point of view. Finally, we will briefly mention the hardware implementation of our system in section 6.

2 Von Neumann States and Transitions

As mentioned before, each cell of the von Neumann CA possesses 29 states. These states can be divided into several different groups with regard to their "functionality".

The first state is the quiescent one. A cell in this state has no influence on its neighborhood. Every unused cell of the cellular array is in the quiescent state.

Then we find the 16 states responsible for the propagation of excitations through the cellular array. Each of these states has a direction, i.e. one of the four cardinal ones, and can be excited or not. Moreover there are two kinds of transmission states that propagate different types of excitations: the ordinary and the special ones. An ordinary (respectively special) transmission state becomes excited if it receives an ordinary (respectively special) or a confluent (see below) excitation on one of its inputs. Moreover, they introduce a delay of one time unit in the propagation of the excitations and act as OR gates.

A third group contains the confluent states. These are also used for the transmission of excitations. A confluent cell becomes excited only if all of the neighboring ordinary transmission states directed to it are excited. It consequently acts as an AND gate. The confluent states are also used to convert an ordinary excitation into a special one and they permit the splitting of a transmission line, acting as fan-outs. Moreover, they introduce a two time units delay in the propagation of information. Finally, they act as memory units: when none of its neighbours is an outgoing transmission state, if the confluent has been previously excited, it keeps the excitation until it can release this latter through a valid transmission state.

Pesavento's extension of the standard rules provides another type of confluent state that permits the crossing of excitations.

The last group contains the sensitive states that are transitory ones and permit the construction of each specific cell type: when a quiescent state receives an excitation, it goes in the first sensitive state. Then, its state changes at each time unit as seen in figure 2.

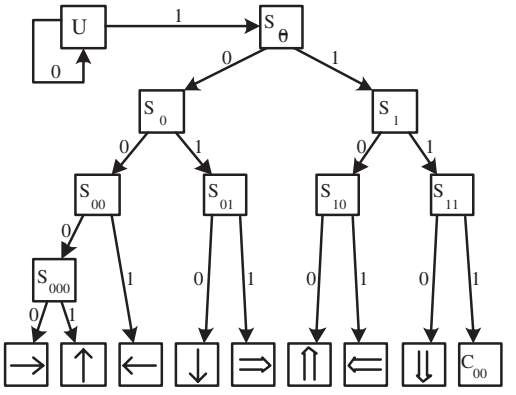

Fig. 2. Sensitive states and construction process

To the construction process corresponds a destructive one: the transition from any state to the quiescent one. When a cell is in an ordinary transmission or in a confluent state and receives a special excitation, it is destructed and becomes quiescent again. The same thing happens when a special transmission state receives an ordinary excitation.

3 Tom Thumb Algorithm

To find an alternative self-replication mechanism for the UC, we define the entire constructor as a two-dimensional *organism* composed of multiple cells. Each cell is defined by two *genes*: the first one corresponds to the cell's state according to von Neumann while the second, as we will see, will be used to direct the construction and replication processes. Using these definitions, we can then apply to the machine a self-replication mechanism known as the Tom Thumb Algorithm (TTA) [4,5].

Without loss of generality, we will present the TTA construction process of a four-cell organism defined by its *genome* showed at the top of figure 3. The numbers correspond to von Neumann states, while the other genes define the directions of the construction path (arrows) as well as the branching directions for the self-replication process (background color of arrows in the figure).

Moreover, although defined by only two genes, each cell of the organism has four memory positions: the two at the right in figure 3 are used to store the cell configuration, i.e. its state and the construction information, while the two remaining ones keep a copy of the genome indefinitely moving around the organism.

The construction process proceeds as follows: when the genome is injected in the cellular array, the first cell receiving it shifts the incoming genes until it has stored its configuration in the corresponding registers (t=4 in figure 3). At that time, the cell knows where to forward the next incoming genome information

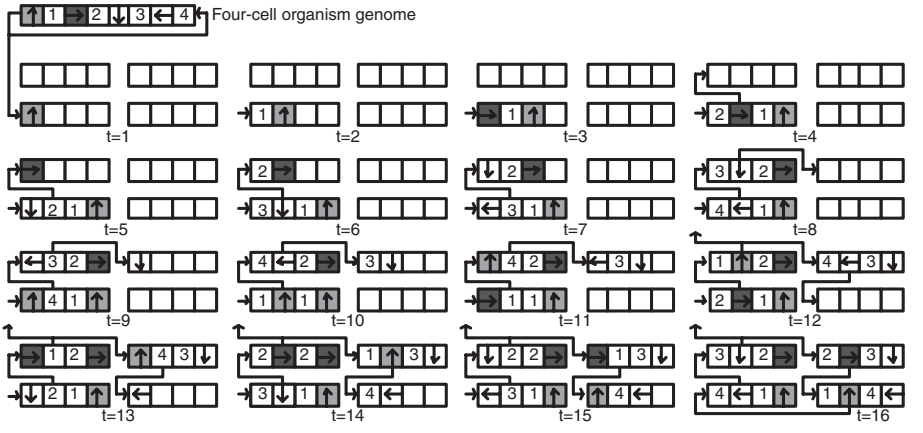

Fig. 3. Genome and first steps of the TTA creation of a four-cell organism

(to the North for the cell of the figure). The next genes arrive in the upper cell which will also shift them to get its configuration and find the way to forward the remaining genes (East, t=8). This process then repeats itself until the organism is fully constructed, i.e. at time t=16 in the figure.

Note that twice the whole genome has been injected for the organism construction: one half of the genes is fixed in the configuration registers while the other half is continuously shifted around the organism, following the construction path.

The self-replication process is started when the first gene of the genome arrives in a cell whose configuration allows branching. In our example, at time t=12, a replication path is consequently initiated to the North. The running genome is duplicated and sent through the replication path. This process obviously starts the construction of a second organism whose genome is the same as for the first one: the organism replicates itself.

Note that when the genome has been transmitted twice through the replication path, the construction of its copy is finished and the branching replication path can be removed. Note also that the construction path of a valid organism has to be a loop; if it were not, the copy of the genome would not have been able to run around the organism and the replication process would have failed.

4 TTA and UC

Our main goal was to add to the von Neumann system an innovative process, in our case a modified version of the TTA, that could drastically reduce the number of cells used for the self-replication. Then, the resulting system had to keep every standard functionality of the original von Neumann CA, as well as implementing the Pesavento crossing ability. Consequently, we have separated

our system design in two different layers and we added a new cell type to the system: the *activator*, used to launch the self-replication process.

The first layer (UClayer) contains the von Neumann standard information coded on four bits. The two first bits determine the cell type (quiescent 00, ordinary arrow 01, special arrow 10, and confluent 11). The two remaining ones define the arrow direction or the confluent subtype (memory, crossing or activator). We show in figure 4 the different UC layer codes and their corresponding cell type.

Type	Dir	Cell symbol	
00	$\emptyset$	U	Quiescent
01	00	$\uparrow$	ordinary North arrow
01	01	$\rightarrow$	ordinary East arrow
01	10	$\downarrow$	ordinary South arrow
01	11	$\leftarrow$	ordinary West arrow
10	00	$\Uparrow$	special North arrow
10	01	$\Rightarrow$	special East arrow
10	10	$\Downarrow$	special South arrow
10	11	$\Leftarrow$	special West arrow
11	00	–	not used
11	01	R	activator confluent
11	10	C_+	crossing confluent
11	11	C_{xx}	memory confluent

Fig. 4. UC layer codes

The second layer (TTAlayer) is only used for the construction and replication processes. It's also defined with four bits: the first two give the direction of the replication path, the third bit is a flag that says if the cell is the first one of the organism and, finally, the last bit corresponds to the end-of-organism cell flag.

Moreover, in our TTA version, the path does not have to be a loop, as in the standard algorithm: in our case, the path begins at the start-of-organism cell and then goes through the entire organism, from cell to cell, until it reaches the end-of-organism cell.

4.1 Cell Construction Process

As we have just seen, a TTA-UC cell is fully defined by the eight configuration bits of the two layers. This fact did not obviously permit us to keep the same construction process as in the original von Neumann system, but we maintained the same general idea: a quiescent cell is transformed in another cell when it receives the corresponding (ordinary or special) excitation. To implement the construction process, as shown in figure 5, we added three flags (with the grey background) to the eight configuration bits.

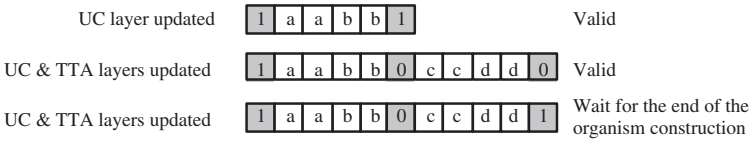

Fig. 5. The different construction pulses

Reading the bits from left to right, the first flag bit is used to start the cell construction process and is always equal to '1'. Then we find the UClayer type on two bits (aa in figure 5), followed by the UClayer direction, on two bits too (bb).

After these, we have the second flag bit: if it is equal to '1', the cell construction process stops. In such a case, the cell has updated its UClayer state, but without modifying its TTAlayer configuration. This could be used to create a new self-replicating organism on a pre-existing TTAlayer, or just to change some cells in a TTA-UC structure without affecting its self-replication ability.

In the other case, i.e. if the second flag is equal to '0', the construction process continues: the two next bits update the TTAlayer direction (cc) and the last two define the TTAlayer type (dd), i.e. start or end of organism.

At the end, we find the last flag bit. While it is equal to '1', it makes the cell wait for the entire organism to be constructed, before being able to act like an usual von Neumann cell. During that time, the cell only forwards every excitation it receives in the direction defined by its TTAlayer.

A cell can then be constructed using a pulse of six excitations, as long as we want to keep the TTAlayer unchanged. Otherwise, the construction pulse needs eleven excitations, with the last being inactive for a single cell creation and active in the case of an organism self-replication.

4.2 Organism Self-Replication

We will now present the self-replication process of a simple organism. For clarity, we will take as example the replication of a two by two cells organism. The two layers of the organism and the configuration bits of its cells (without the construction flags) are shown in figure 6. The start-of-organism cell is in dark grey while the end-of-organism one is in light grey.

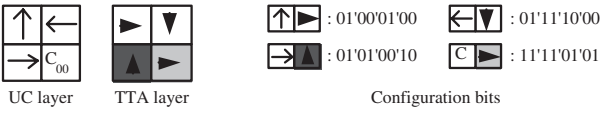

Fig. 6. Two layers and configuration bits of the organism

We will first present the emission of the organism genome and then, show how this genome is used in the construction of the new replicated organism.

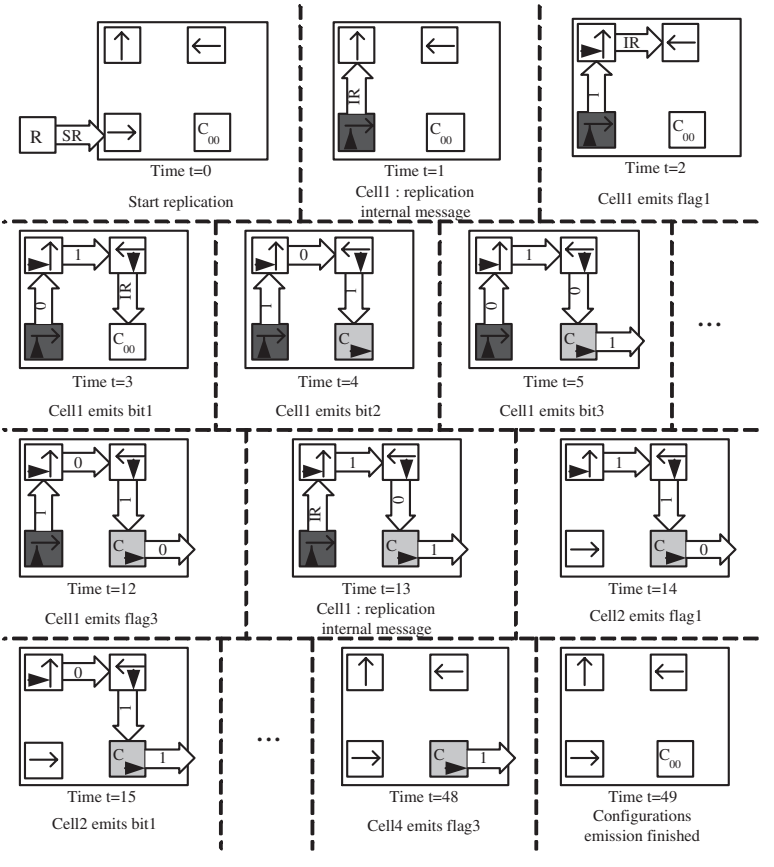

Fig. 7. Some steps of the TTA-UC genome emission

Genome emission: An organism will start its replication process if one of the neighbours of the start-of-organism cell is an excited activator cell (R in figure 7 at time t=0). In such a case, this latter sends a StartReplication message (SR in the figure); the start-of-organism cell then emits to its TTA direction a InternalReplication message (IR) at time t=1. This latter will then be forwarded along the TTA path through the entire organism until it reaches its last cell, making every cell of the organism become in a TTA forwarding state: from that time, they transmit to their TTA direction every excitation they receive.

Just after the InternalReplication message, the start-of-organism cell sends its configuration bits with the appropriate flags during the following 11 clock steps.

As for the first message, these configuration bits are then forwarded through the TTA path until they reach the end-of-organism cell. This latter transmit the configuration pulse to the UC layer of the next cell in its TTA direction.

When the first cell has finished sending its configuration stream, it emits once again a InternalReplication message (t=13) meaning it has finished its replication task. When the next cell on the TTA path receives this InternalReplication message, it goes out of the TTA forwarding state and begin the emission of its own configuration bits.

This process repeats itself in a similar way until the end-of-organism cell has transmitted its last configuration bit: at that time (t=49), the organism has finished its genome emission.

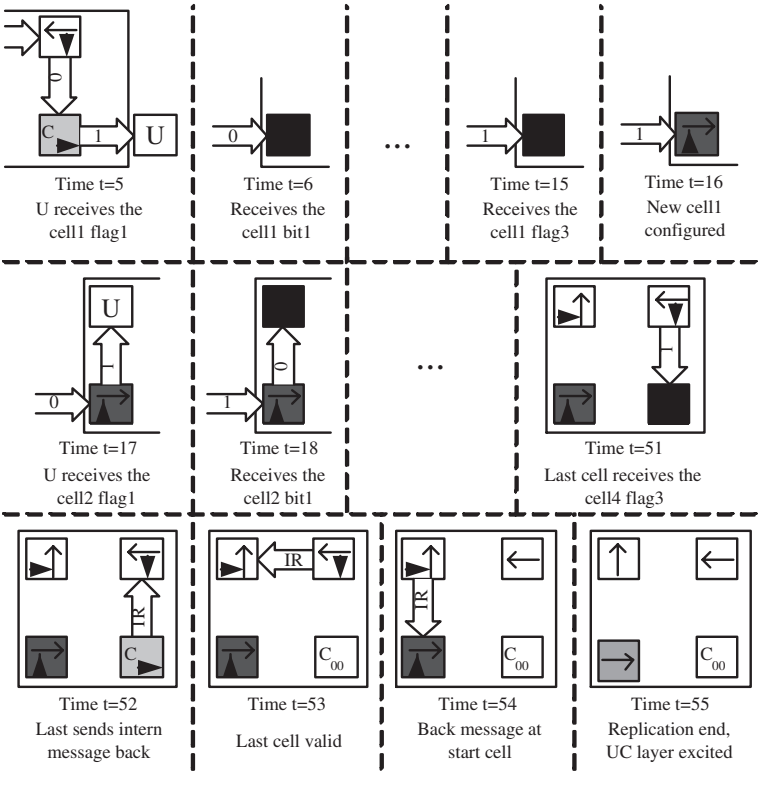

Fig. 8. Some steps of the TTA-UC replicate construction

Construction: We said that the end-of-organism cell transmits the configuration pulses to the UC layer. The excitations then consequently follow the UC layer specifications: they can travel along an ordinary arrows path, if any, until they reach a quiescent cell to configure. In figure 8, that happens at time t=5.

When a quiescent cell has received eleven building excitations, it is fully configured. But, as we have seen, the last configuration bit has to be chosen equal to '1' in the self-replication process. This flag implies that the cell must stay in a TTA forwarding state until the end of the organism construction. Indeed, it has to transmit the next received excitations to its TTA direction in order to pursue the creation of the next organism cells, in the good direction.

When the last configuration bit has reached the new end-of-organism cell (t=51), this latter sends an InternalReplication message back along the TTA path. This message, that notifies the end of the organism construction is forwarded back by each cell of the organism until it reaches the new start-of-organism cell, at t=54 in the figure.

The next time unit, the start-of-organism cell becomes excited in its UC layer and can launch the classical UC behaviour of the new organism: the self-replication process is terminated.

5 TTA-UC and Evolution

As explained before, and unlike the standard Tom Thumb Algorithm, our TTA-UC system does not possess a running genome that keeps unchanged the initial configuration of the organism. In fact, our replication system works by self-inspection [6]: the organism generates the bits configuration that permits to copy its current state (and not the initial one).

Having such an ability should permit a TTA-UC organism to evolve through its replication process: for example if a cell of the organism has got its UCconfiguration modified during its normal behavior, without changing its TTAlayer information, then the organism can still replicate itself but will also copy the modified cell. As a result, this new organism will not be a perfect copy of the initial one: almost every UClayer modification can propagate through the organism generations. An organism population could consequently evolve by the integration of the acquired modifications into the organism configuration of the 'new generations'.

6 Hardware Implementation

The hardware implementation of the TTA-UC uses the BioWall, a two- dimensional electronic wall designed for bio-inspired applications [7]. In order to test the concept, we have realized a simple von Neumann pulser composed of about thirty cells. We were able to verify that the pulser can self-replicate in addition to realizing its usual functionality.

Moreover, due to the huge number of states the TTA-UC design would have used for a standard CA implementation, we realized the system following the Data and Signal Cellular Automaton (DSCA) concept [8,9]. Such a choice permited us to design the TTA-UC almost intuitively and to make the design fit the limited place at disposal.

7 Conclusion

The designed system comes from the unification of the Tom Thumb Algorithm with the standard von Neumann architecture. Its major quality is the great diminution in the number of cells that an organism needs for its self-replication. In fact, with this new TTA-UC system, any von Neumann organism can now be configured in order to possess the self-replication ability. As a result, the concept of Universal Constructor in itself becomes obsolete.

Moreover, we believe that our TTA-UC system is able to emulate all of the precedent von Neumann functionalities. A further interesting work could be to confirm that a Turing Machine can be successfully emulated by our new TTA-UC architecture. Additionally, one should be able to realize this implementation with only small modifications to the original design of the von Neumann Turing Machine, due to the differences in the cell construction process.

References

1. J. von Neumann. Theory of Self-Reproducing Automata. University of Illinois Press, 1996. Edited and completed by A. W. Burks.
2. W.R. Buckley. On the complete specification of a von Neumann 29-state self-replicating cellular automaton. Private communication, wrb@wrbuckley.com, 2004.
3. U. Pesavento. An Implementation of von Neumann's Self-Reproducing Machine. Artificial Life 2(4): pp. 337-354, 1995.
4. E. Petraglio: Fault Tolerant Self-Replicating Systems. PhD Thesis n^o 2973, EPFL Lausanne, pp. 79-88, 2004.
5. D. Mange, A. Stauffer, E. Petraglio, G. Tempesti. Self-replicating loop with universal construction. Physica D 191, pp. 178-192, 2004
6. Ibàñez J., Anabitarte D., Azpeitia I., Barrera O., Barrutieta A., Blanco H., and Echarte F. Self-inspection based reproduction in cellular automata. Proc. 3rd Eur. Conf. on Artificial Life (ECAL95), LNCS 929, Springer Berlin, pp. 564-576, 1995.
7. G. Tempesti, D. Mange, A. Stauffer, and C. Teuscher. The BioWall: An Electronic Tissue for Prototyping Bio-Inspired Systems. Proceedings of the 2002 NASA/DOD Workshop Conference on Evolvable Hardware, pp. 221-230, 2002.
8. A. Stauffer and M. Sipper. The Data and Signals Cellular Automaton and its application to growing structures. Artificial Life 10(4): pp. 463-477, 2004.
9. A. Stauffer and M. Sipper. Data and signals: A new kind of cellular automaton for growing systems. Proceedings of the 2003 NASA/DOD Conference on Evolvable Hardware, Los Alamitos CA, pp. 235-241, 2003.