HAL
open science

An Upper Bound on the Number of States for a Strongly Universal Hyperbolic Cellular Automaton on the Pentagrid

Maurice Margenstern

▶ To cite this version:

Maurice Margenstern. An Upper Bound on the Number of States for a Strongly Universal Hyperbolic Cellular Automaton on the Pentagrid. Journées Automates Cellulaires 2010, Dec 2010, Turku, Finland. pp.168-179. hal-00541965

HAL Id: hal-00541965
https://hal.science/hal-00541965v1

Submitted on 1 Dec 2010

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Journées Automates Cellulaires 2010 (Turku), pp. 168-179

AN UPPER BOUND ON THE NUMBER OF STATES FOR A STRONGLY UNIVERSAL HYPERBOLIC CELLULAR AUTOMATON ON THE PENTAGRID

MAURICE MARGENSTERN

Université Paul Verlaine, LITA EA 3097, UFR MIM and CNRS, LORIA, Campus du Saulcy, 57045 Metz Cedex 1, France
E-mail address: margens@univ-metz.fr
URL: http://www.lita.sciences.univ-metz.fr/~margens

Abstract. In this paper, following the way opened by a previous paper deposited on *arXiv*, see[7], we give an upper bound to the number of states for a hyperbolic cellular automaton in the pentagrid. Indeed, we prove that there is a hyperbolic cellular automaton which is rotation invariant and whose halting problem is undecidable and which has 9 states.

1. Introduction

In [7], we gave a general tool to embed a 1D-cellular automaton into a whole family of tilings of the hyperbolic plane and into two tilings of the hyperbolic 3D-space.

In this paper, we try to improve the method in order to find a strongly universal cellular automaton in the pentagrid. We remind the reader that by strong universality, we mean a cellular automaton which mimics the computation of universal devices starting from a **finite** configuration. We also remind the reader that the pentagrid is the tiling $\{5, 4\}$, *i.e.* the tessellation of the hyperbolic plane based on the regular pentagon with right angles, see [2, 4]. Within the limit on the number of pages given to the author, the paper cannot be self-contained. This is why we assume that the reader is familiar with both cellular automata and their implementation in tessellations of the hyperbolic plane, and we refer him/her to the above references and to this additional one, [5], in case he/she would not be familiar with these notions.

In Section 2, we give the outline of the construction. In Section 3 we implement the preliminary structure of the implementation. In Section 4 we give a construction with 13 states. In Section 5, we reduce this number by one state, which will give the way to Section 6 where the number of states is reduced to 9 of them. Section 7 concludes the paper with indications on further work.

Key words and phrases: cellular automata, strong universality, hyperbolic spaces, tilings.

creative commons

2. Scenario

In traditional literature on cellular automata, a **quiescent** state is defined as a state q such that if a cell and all its neighbours are under state q, the cell remains under state q at the next time of the clock. By analogy with the empty squares of the tape of a Turing machine which are said to contain the **blank** symbol, we shall fix a quiescent state and we shall call it the **blank**.

For cellular automata which have a blank state, an initial finite configuration is a configuration in which all cells are blank except, possibly, finitely many of them. It is plain that if the initial configuration is finite, all further configurations are also finite, even when the computation requires an infinite time: in this case, there may be no uniform bound to the size of each configuration.

Let us now turn to the idea of the construction.

The idea of [7], which embeds any 1D-cellular automaton with infinite initial configuration is very simple: it consists in embedding the 1D-structure of the considered cellular automaton into the desired tiling of the hyperbolic plane. In this construction, we simply had to devise a simple way to make the cells of the embedded structure different from the other cells of the tiling. In this way, we can easily transport the rules of the 1D-cellular automaton into those of the hyperbolic cellular automaton. The key point is that in this construction, the differentiation is made a priori: it is given in the initial configuration.

If we wish to implement a 1D-cellular automaton which starts its computation from a finite configuration, then we have to go on the embedding at the same time as the computation is going on. And so, we have to find out a simple way to construct the 1D-structure together with the computation. But this is not enough. Remember that the halting of the computation of a cellular automaton is defined by the occurrence of two consecutive identical configurations. And so, when the computation of the 1D-cellular automaton is completed, we have to stop the construction of the 1D-structure.

In [3], the propagation of the tree structure of the pentagrid is implemented in a triangular cellular automaton, and this automaton can easily be adapted to the pentagrid, also as a **rotation invariant** one. Rotation invariance means that the new state is unchanged if we perform a circular permutation on the neighbours of the cell, this cell being excepted. This was done in [5] and repeated, as an example in [6], not in the shortest way as in that context the goal was that a cell should recognize whether it is black or white with respect to the Fibonacci structure simply by looking at its neighbourhood. Here, we do not really bother of this condition so that instead of six states as it is the case in [5], three states are enough: the blank, a white one and a black one. As we have the choice for the place of the initial configuration, we can place it around the central cell which spares us the burden of initializing the propagation of the structure: it is enough to assume it is installed in the initial configuration and to continue it, this spares one state.

However, we have to stop the computation, which means that a signal has to be sent to stop the others. In order to perform this task, it is needed to slow down the propagation. Indeed, the speed of a signal is at most 1. And so, if the halting signal travels at speed 1, it can catch up previously sent signals only if these latter signals travelled at a lower pace. Now, slowing down necessarily costs states as we shall see. But, fortunately, the 1D-cellular automaton which we shall consider is also slow, so that we shall not have to slow down too much.

After that, we have to look at the way to find a 1D-cellular automaton which is strongly universal with a small number of states. We shall use the implementation of the 7×4 universal Turing Machine of Marvey Minsky which is precisely described in [1]. This gives us a 1D-, strongly universal cellular automaton with 7 states. In the rest of the paper, we denote this cellular automaton by $\mathcal{L}$.

Now, to see how many states we can obtain, we have to go into finer details of the implementation. The first step is the propagation of the 1D-structure which is a common feature of the three cellular automata which we construct in the paper.

3. Implementation of the 1D-structure

In [7], in the case of the pentagrid, we implemented the line of a 1D-cellular automaton along a line of the pentagrid. We remind the reader that such a line is any line which contains a side of a pentagon of the tiling: such a line contains the sides of infinitely many pentagons which can be gathered into two sequences of pentagons indexed by $\mathbb{Z}$, two consecutive pentagons having a side on the line, these just indicated sides being also consecutive.

Here, as we start from a finite configuration, we have at most finitely many cells along such a line and our task is to devise a way to go on this line as a continuation of the segment which already exists.

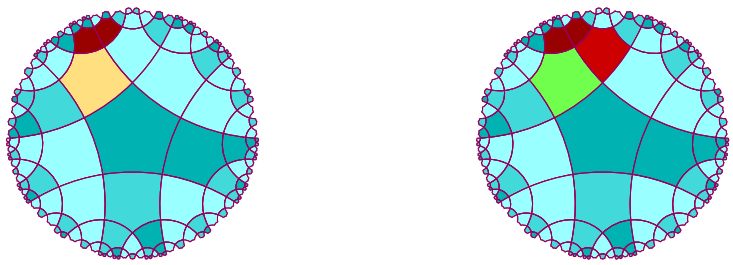

Figure 1: The first two configurations of the propagation of the $1D$-structure. Left-hand side: initial configuration, say time 0. In dark red, cells in state $\mathbf{B}$, in light yellow, the cell in state $\mathbf{W}_0$. Right-hand side: time 1. In bright red, the cell in state $\mathbf{B}_0$. In green, the cell in state $\mathbf{W}_1$. The blue cells represent the blank denoted by $\mathbf{N}$: the different hues of blue remind the tree structure of the tiling, but they represent a single state. Note that here, the central cell is not the central pentagon of the figure, it is the dark red cell in contact with the light yellow, green one in the left-, right-hand side picture respectively.

Now, the problem can be a bit simplified by the fact that $\mathcal{L}$ possesses an interesting feature. The cellular automaton $\mathcal{L}$ implements a Turing machine which is an interpreter of tag systems. This mean that we may assume that the Turing machine works on a semi-infinite tape: *i.e.* the tape has an end but it is infinite in one direction only. This is particularly interesting for our implementation: this allows us to implement a ray only, so that we can put the end of this ray around the central cell. For the propagation algorithm, we can define the first two configurations as illustrated by Figure 3.

We need six states for the propagation of the ray: $\mathbf{N}$, $\mathbf{B}_0$, $\mathbf{B}$, $\mathbf{W}_0$, $\mathbf{W}_1$ and $\mathbf{W}$. Informally, cells in $\mathbf{B}$ will follow a branch of black nodes of one of the Fibonacci

trees rooted around the central cell. This branch is the leftmost branch of the chosen Fibonacci tree. Now, the cells **W** follow the rightmost branch of the next Fibonacci tree while clock-wise turning around the central cell. It is easy to remark that the pentagons of these two branches share a ray which supports one side of each one of these pentagons.

The propagation itself advances at a speed 1/2. This is suggested by the presence of the states $\mathbf{B}_0$ and **B** as well as by that of the states $\mathbf{W}_0$ and $\mathbf{W}_1$.

The mechanism is the following. A new **B** is produced by the transformation of $\mathbf{B}_0$ into **B**. Now, a new $\mathbf{B}_0$ is obtained by the continuation of both the black branch and the white one. It is created by the simultaneous occurrence of a **B** and a $\mathbf{W}_0$ around a blank cell abutting the cell at contiguous sides and in a precise order: while counter-clockwise turning around the cell, we first meet **B** and then, immediately, $\mathbf{W}_0$. The three cells, the blank, **B** and $\mathbf{W}_0$ share a common vertex. We know that there is a fourth cell. In the initial configuration it is a cell in **B**. In the other configurations, when this local configuration occurs, the fourth cell is a cell in **W**: it is a cell in $\mathbf{W}_1$ which evolved into **W**. The roles between $\mathbf{W}_0$ and $\mathbf{W}_1$

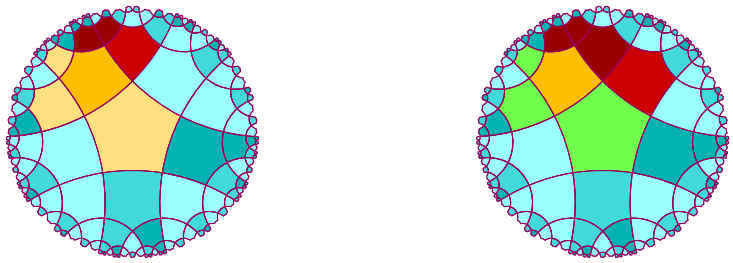

Figure 2: Left-hand side: time 2 of the propagation. Right-hand side: time 3 of the propagation. In bright yellow, state **W** which indicates a fixed cell. The blue cells have the same meaning as in Figure 3.

are the following. A cell in $\mathbf{W}_0$ becomes **N** at the next time if it is surrounded by cells in **N**. Otherwise, it becomes $\mathbf{W}_1$. Now, a cell in $\mathbf{W}_1$ becomes **W** if and only if it sees a neighbour in $\mathbf{B}_0$. Otherwise, it becomes **N**. Now, a cell in **N** becomes $\mathbf{W}_0$ if and only if it has one neighbour in $\mathbf{W}_1$ exactly. Otherwise, it remains **N**.

Now, we can see that the configuration which allows a cell in **N** to become $\mathbf{B}_0$ requires the cells in **W** or $\mathbf{W}_0$ to be one step in advance with respect to those in **B** and $\mathbf{B}_0$. This is also allowed by the progression of the cells in $\mathbf{W}_0$. The condition on $\mathbf{W}_0$ allows us to stop the progression of the $\mathbf{W}_0$'s which do not follow the ray. Whence the importance on the condition of the transformation of the $\mathbf{W}_0$'s and also of the $\mathbf{W}_1$'s which disappear unless they can see a cell in $\mathbf{B}_0$, in which case they become **W**.

We have no room here for the table of the rules. Such tables can be found in [8]. We refer the reader to this paper for them. In the present paper, the expression *the rules for* $\mathcal{A}$ refers to the tables in [8] which display the rules of the automaton $\mathcal{A}$. Note that in all the tables of [8], there are two kinds of rules. In the first group, the current state of the cell is left unchanged: this is why these rules are called **conservative**. In the second group, the current state of the cell is changed: these rules are called **propagation** rules.

Basically, there are two propagation rules: the rule `N B W0 N N N B0` and the rule `N W1 N N N N W0`. In both of them, the blank is changed into a cell which will

contribute to the extension of the 1D-structure. Now, the other rules contribute to create the context required by these two propagation rules as well as the transformation of a first signal, $\mathbf{B}_0$ and $\mathbf{W}_0$ into the final one, $\mathbf{B}$ and $\mathbf{W}$, respectively.

The fact that we have three signals for the white node instead of two as the speed of progression of the $1D$-structure is $\frac{1}{2}$ is explained by the structure of the pentagrid: when a cell in $\mathbf{W}_1$ propagates the signal $\mathbf{W}_0$, this is performed upon several cells, at least two of them, while it is needed for only one of them. This is the reason of the signal $\mathbf{W}_1$ which is an intermediate step between $\mathbf{W}_0$ and $\mathbf{W}$. Now, in order to keep the speed $\frac{1}{2}$, the alternation is performed between $\mathbf{W}_1$ and $\mathbf{W}_0$.

It is now time to go to the other parts of the implementation. In these parts, we shall refer to the cells in $\mathbf{B}$ and in $\mathbf{W}$ of the just described construction as the **ray** in which the cells in $\mathbf{B}$ constitute the **track** and the cells in $\mathbf{W}$ constitute the **support** of the track.

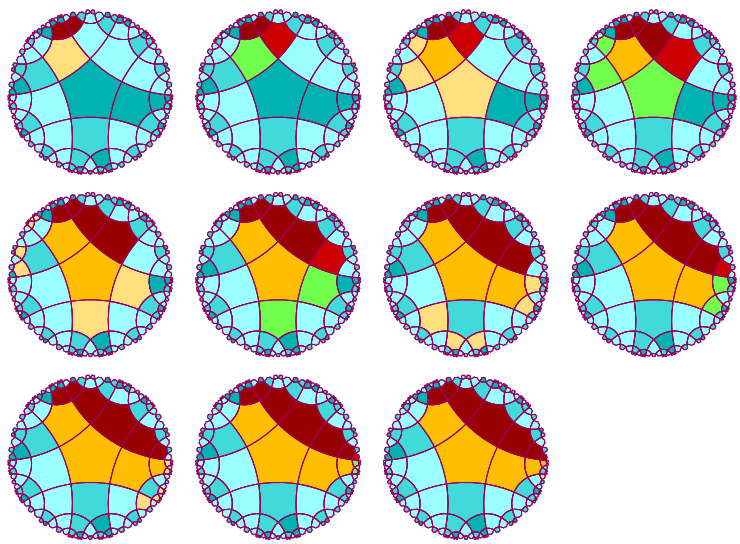

Figure 3: Illustration of the propagation from the beginning until a few steps. In bright yellow, state $\mathbf{W}$ which indicates a fixed cell. The blue cells have the same meaning as in Figure 3.

4. The 13-state cellular automaton

First of all, we remind the reader that in this paper, we shall consider deterministic cellular automata only, as $\mathcal{L}$ is itself a deterministic $1D$-cellular automaton. Let us denote by $\mathcal{A}$ the automaton which implements the computation performed by $\mathcal{L}$. We require $\mathcal{A}$ to be rotation invariant.

In the previous propagation, we consider that $\mathbf{B}$ represents the blank of $\mathcal{L}$ which has to be distinct from the blank $\mathbf{N}$ of $\mathcal{A}$. Indeed, if we give the same state for the two blanks, we shall have problems with the propagation, as can easily be seen from the scenario of Section 2.

An important point is that in the working of $\mathcal{L}$, it can be noticed from [1] that the configuration of $\mathcal{L}$, *i.e.* the smallest interval which contains the non blank

cells, remains the same during at least three consecutive steps and that it may go outside by one cell, only at the fourth time. This means that the progression of the configuration of $\mathcal{L}$ is much slower than the propagation of the ray described in Section 3. As a consequence, when the signal of $\mathcal{L}$ goes outside the current configuration, the track is ready to deliver free blank cells.

Accordingly, the computation of $\mathcal{A}$ is able to perform that of $\mathcal{L}$ during the propagation stage. In fact, it is enough to consider that in the process described in section 3, $\mathbf{B}$ can be any of the states of $\mathcal{L}$ and that it must be the blank for the cell in $\mathbf{B}_0$ which is transformed into $\mathbf{B}$. This latter $\mathbf{B}$, at this moment, is the blank of $\mathcal{L}$. Indeed, the cells of the track are the single one which, except the ones which are at the ends of the track, have two neighbours which are also cells of the track. In fact, except for two exceptional cells, a rule of the track is of the form $y\ x\ \mathtt{W}\ z\ \mathtt{N}\ \mathtt{N}\ u$ where $xyz \to u$ is a rule of $\mathcal{L}$. For the exceptional cells, the origin and its neighbour of the track, the cell has three neighbours under $\mathbf{N}$, and the origin has a single neighbour on the track. All these features can easily be seen from the pictures of Figure 3, there cannot be ambiguity about these local configurations.

This means that, presently, as $\mathbf{B}$ is one of the states of $\mathcal{L}$, $\mathcal{A}$ has 12 states as $\mathcal{L}$ itself has 7 states, see [1].

Now, let us closer look at the working of $\mathcal{L}$. In [1], $\mathcal{L}$ mimics rather closely the computation of Minsky's Turing machine. In particular, there is a state T, notation of [1], which represents the position of the head of the machine. In the simulation devised in [1], the halting is performed by the disappearance of T. Our task is to change this transformation into a signal which will trigger the stage at the end of which the computation of $\mathcal{A}$ will also halt in the traditional sense of the halting of a cellular automaton starting from a finite configuration, see Section 2.

To perform this task, we replace the instruction $0Ty \to 0$ of the table of $\mathcal{L}$ in [1] by the instruction $OTy \to \mathbf{H}$, where $\mathbf{H}$ is a new state of $\mathcal{A}$. Also, we append the new instructions $\mathbf{H}y\mathbf{B} \to \mathbf{B}$, $\mathbf{B}0T \to \mathbf{B}$, where $\mathbf{B}$ is in both cases $\sqcup$, the blank of $\mathcal{L}$. Note that $\sqcup 0\, T \to \sqcup$ is a rule of $\mathcal{L}$, see [1].

Now, we can make a bit more precise the scenario depicted in Section 2. As just indicated, $\mathbf{H}$ appears on the track. In some sense it is far from the ends as we can put several blanks to the left of the leftmost non blank cell of the Turing tape in the initial configuration. We remind the reader that we may choose the initial configuration and we may choose it so that the initial segment of $\mathcal{L}$ outside which there are only blank cells is in middle of the initial configuration, with at least two blank cells outside this segment. We decide that $\mathbf{H}$ does not affect the cells of the track which will remain unchanged. However, we decide that a cell in $\mathbf{W}$ which sees at least one $\mathbf{H}$ among its neighbours becomes $\mathbf{H}$ itself.

In this way, the cells of the support of the track are progressively changed to $\mathbf{H}$. The corresponding rules are given in [8] and they are illustrated by Figure 4. We have just to see that we can effectively stop the generation of cells in $\mathbf{W}_0$ and in $\mathbf{W}_1$, which will to its turn stop the production of $\mathbf{B}_0$. We also have to check that no problem arises on the fixed end of the ray which represents the leftmost part of the Turing tape.

Figure 4 shows how the propagation of the ray is stopped by the arrival of the states $\mathbf{H}$. For simplicity, call signal $\mathbf{H}$, the propagation of $\mathbf{H}$ replacing the state $\mathbf{W}$ in the cells of the support of the track.

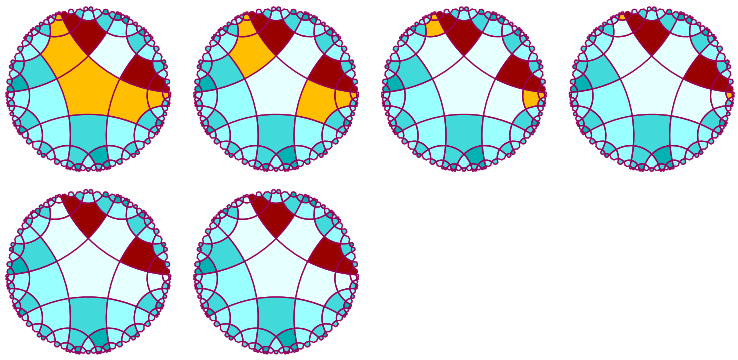

Figure 4: The propagation of the **H**-signal. Note that the back ground of blue cells is the same as in the previous figures.

First, assume that the signal **H** arrives as indicated in the figure: almost all cells of this part of the support are now in state **H**, and just a single cell in **W** remains whose next neighbour is a cell in $\mathbf{W}_1$. Necessarily, the cell in **W** is changed to **H** and the cell in $\mathbf{W}_1$ becomes **W**: the cell in $\mathbf{W}_1$ cannot see what is on another side of its neighbour in **W**. And so, at the next step, we have a cell in **W** again for which one neighbour is in **H** and two others are in $\mathbf{W}_0$. At the next time, the cells in $\mathbf{W}_0$ become $\mathbf{W}_1$ and the cell in **W** becomes **H**. Now, the cell in **H** is neighbouring a cell in $\mathbf{W}_1$. If the cell in $\mathbf{W}_1$ turns to **W** again, then the signal will never stop the propagation of the ray. And so, the cell in $\mathbf{W}_1$ turns to **H** when one of its neighbours is in **H**: this is possible as no rule with the current state $\mathbf{W}_1$ involved a neighbour in **H**. And this solves the problem: at the next time, the cells in $\mathbf{W}_0$ either have their five neighbours in **N**, or they have a neighbour in **H**. We decide that the neighbouring of **H** makes a cell in $\mathbf{W}_0$ to turn to **N** too, and this stops the process. It can be checked that the rules of [8] allow to perform this task. This was done by the computer program which also computed the data for the PostScript file producing Figures 3, 4, 4 and 4. The computer program also checked the rotation invariance of the rules.

From the figure, it is not difficult to see that the situation illustrated by Figure 4 is general: as the signal goes faster than the progression of the ray, there will always be a time when the rightmost cell in **H** will be close to the rightmost cell in **W** at a time when this cell is neighboured by cells in $\mathbf{W}_1$. Indeed, if there are two cells in **W** between the cell in **H** and the cell in $\mathbf{W}_1$, at the next time, the cell in$\mathbf{W}_1$ which is close to the track becomes **W** while the others become **N**, and the blank cells close to $\mathbf{W}_1$ become $\mathbf{W}_0$. Now, between the rightmost cell in **H** and the cell in $\mathbf{W}_0$, there are two cells in **W**. But at the next time, the cell in **W** close to **H** becomes **H** and the cell in $\mathbf{W}_0$ which is close to the border becomes $\mathbf{W}_1$. And so, at this time, we have the same configuration as the one illustrated by Figure 4. This proves that, in all cases, the signal **H** stops the progression of the ray as this was planned.

We remain with checking that the progression of the signal **H** in the other direction is stopped by the origin: this is illustrated by Figure 4. See the rules in [8].

At this point, we have proved the following result:

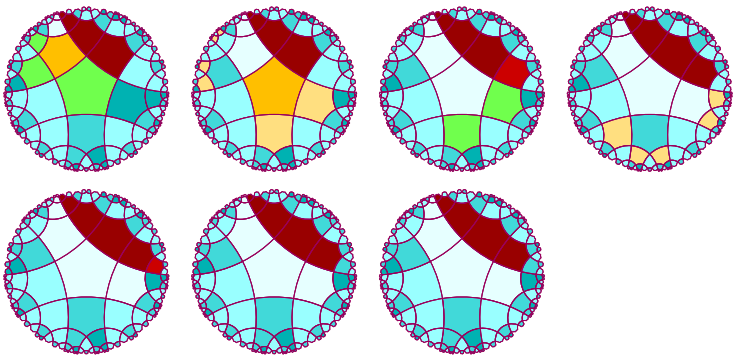

Figure 5: How the **H**-signal stops the propagation of the ray.

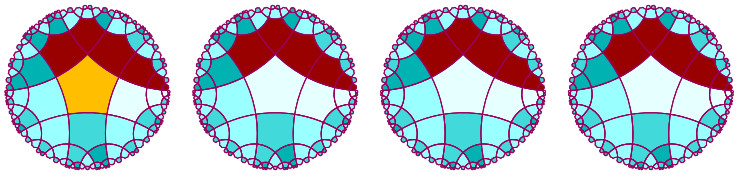

Figure 6: How the **H**-signal is stopped at the other end of the ray, near its origin.

Theorem 4.1. *There is a rotation invariant hyperbolic cellular automaton in the pentagrid which starts from finite configurations and whose halting problem is undecidable which has* 13 *states, the blank included.*

Indeed, $\mathcal{A}$ satisfies the statement of Theorem 4.1. However, the reader may wonder why we stated that the halting problem of $\mathcal{A}$ is undecidable and why we did not state that $\mathcal{A}$ is strongly universal? This is due to a strange property of Minsky's Turing machine with 7 states and 4 letters, a property which is inherited by $\mathcal{L}$ as it closely simulates this Turing machine. The problem lies in the way the Turing machine detects the halting of the simulated tag system. In fact, in this machine, when the halting production is found, the Turing machine erases its tape so that when it stops, the content of the tape can no more be red. This 'defect' of Minsky's machine was noticed and corrected by Rogozhin in [11]. Of course, 'morally' the machine is universal but rigorously, we can say no more than the statement of the theorem.

5. The 12-state cellular automaton

Now, we can show that a slight tuning of $\mathcal{A}$ allows us to obtain a cellular automaton which simulates the computation of $\mathcal{L}$ using 12 states only.

The idea is to replace the state **H** by one of the states of $\mathcal{A}$ which is not used by $\mathcal{L}$. In fact we have no choice. State **W** cannot be chosen as the support of the track consists already of cells in **W**. Using **W** would require to circumvent the support which would lead to more states. Similarly, neither $\mathbf{W}_0$ nor $\mathbf{W}_1$ can be used as they contribute to continue the propagation. For the same reason, $\mathbf{B}_0$ is rules out and, of course, **B** cannot be used: this does not give a clear signal that the computation halted. And so, the only possibility is **N**.

Let $\mathcal{B}$ be the cellular automaton in the pentagrid obtained from $\mathcal{A}$ by replacing **H** by **N** in the rules for $\mathcal{A}$.

We can easily see that under this replacement of **H** by **N** in the rules of [8], the rule obtained from `N H N N N W1 N` is in conflict with the rule we get from `N W1 N N N N WO`, as we require our cellular automaton to be rotation invariant. Of course, the solution is to cancel the rule obtained from `N H N N N W1 N`. And it works: there are no more conflicts, we just have a few repetitions with the rules for $\mathcal{A}$ where **H** does not occur.

And so, we proved the following result:

Theorem 5.1. *There is a rotation invariant hyperbolic cellular automaton in the pentagrid which starts from finite configurations and whose halting problem is undecidable which has* 12 *states exactly, the blank included.*

Note that the remark about strong universality for Theorem 4.1 also holds for this one.

Replacing **H** by **N** boils down to erase the support of the track so that, at the end of the computations, we remain with the track only. This erasing occurs at both ends of the ray. We have no room to provide the figures proving this point. These figures can be found in [8]. They also show that the erasing process stops the propagation of the ray.

Now, could we reduce again the number of states, using the same $1D$ cellular automaton? The next Section gives a positive answer to this question.

6. The 9-state cellular automaton

We have already seen that to reduce the number of states from 13 down to 12, we replaced the state **H** by the state **N**. We can try to go further in this direction. In [7], we reduced the number of states for a weakly universal cellular automaton from 3 states down to 2 ones by replacing the extra state in the simulation with 3 states by a state of the embedded 2-states cellular automaton. So that here, a natural idea is to replace as many as we can states from **N**, **W**, $\mathbf{W}_0$, $\mathbf{W}_1$ and $\mathbf{B}_0$, by states of $\mathcal{L}$ only. If we look at the rules displayed in [1], we can notice that the symbol T has a rather empty sub-table. In particular, there is no rule assigned to TTT, so that we can decide that $TTT \to T$ is used by our new automaton, $\mathcal{C}$. Inspired by the table of the rules given in [1], we shall see that **W**, $\mathbf{W}_0$ and $\mathbf{B}_0$ can be replaced by T, 0 and A respectively. It is enough to check that performing these replacements and taking the above figures, we obtain new rules which are rotation invariant and compatible.

Indeed, consider the first stage of the working of $\mathcal{B}$ which consists in propagating the structure at the same time when the computation is going on. It will be enough to ensure that a pure propagation process can be performed under the new set of states and that there is no contradiction between the new involved rules and the rules derived from the computation of $\mathcal{L}$. We also have to check that the new rules do not disturb the computation itself.

Now, this latter condition entails that we cannot replace $\mathbf{W}_1$ by a state of $\mathcal{L}$. Indeed, imagine that $\mathbf{W}_1$ is a state of $\mathcal{L}$, say α. There are occurrences of α on the track. Each cell of the track has at least two neighbours in state **N**. Consider one of these neighbours. It has α as a neighbour and all the others are in **N**. From Section 3, we know that in this case, the state **N** is replaced by $\mathbf{W}_0$ and this $\mathbf{W}_0$,

as it is not surrounded by cells in $\mathbf{N}$ only, will become $\mathbf{W}_1$. Now, if $\mathbf{W}_1$ has a neighbour in A, which may happen in the track, this $\mathbf{W}_1$ becomes T which will disturb the computation. Even in the case it will not disturb the computation, the production of $\mathbf{W}_0$ nearby the track will be repeated periodically even when the computation has stopped. So that we cannot replace $\mathbf{W}_1$ by a state of $\mathcal{L}$. Can we replace $\mathbf{W}_0$ by a state of $\mathcal{L}$? The answer is yes: in the propagation context, a cell in $\mathbf{W}_0$ has at least four neighbours in $\mathbf{N}$. A cell of the track which is not the blank has two neighbours in $\mathbf{N}$ exactly, so that it is possible to distinguish the role of a distinguished α, depending on its neighbourhood. Moreover, a neighbour in $\mathbf{N}$ of a cell in α would remain unchanged due to the rules $\mathbf{N}\ \mathbf{N}\ \mathbf{N}\ \mathbf{N}\ \mathbf{N}\ \mathbf{W}_0\ \mathbf{N}$ and $\mathbf{N}\ \mathbf{N}\ \mathbf{N}\ \mathbf{N}\ \mathbf{N}\ \mathbf{B}\ \mathbf{N}$.

6.1. The propagation of the ray

Let us have a new inspection of Figure 3. The tables of [8] indicate the rules applied for going from one picture of the figure to the next one. As can be seen from the table, after time 7, no new rule is needed for the propagation of the ray. We can notice that these rules are obtained from those defined for the automaton $\mathcal{A}$, see [8], by replacing $\mathbf{W}$, $\mathbf{W}_0$ and $\mathbf{B}_0$ by T, 0 and A respectively.

However, we have to keep in mind that, during the propagation, the computation is going on. And so, we have to look at the cells of the track and of their neighbours in order to check that they are compatible with the rules for $\mathcal{C}$.

A rule which applies to a cell of the track is of the form

$$\mathbf{BNNB}\ T\ \mathbf{BB} \qquad (*).$$

As we know, $\mathbf{B}$ is a generic name for the states of $\mathcal{L}$. If we replace $\mathbf{B}$ by the states of $\mathcal{L}$, we get rules of the form

$$\alpha_0\mathbf{NN}\alpha_{-1}T\alpha_1\alpha_0^1 \qquad (**)$$

where $\alpha_{-1}\alpha_0\alpha_1 \to \alpha_0^1$ is a rule of $\mathcal{L}$. When $\alpha_0 \in \{B, y\}$, then the rule cannot be confused with one of $\mathcal{C}$ which we have already defined. At the times up to 5, when $\mathbf{B}$ is the current state, it is always $\sqcup$, the blank of $\mathcal{L}$. Now the rules for $\mathcal{A}$, where the current state is $\mathbf{B}$ and which are used up to time 6, either contain at least three consecutive occurrences of $\mathbf{N}$, or they contain an occurrence of A and so, they cannot be confused with a rule $(**)$ whose current state would be $\sqcup$. We have to look at all the other possibilities.

Consider the case when $\mathbf{B}$ is T: the rules for $\mathcal{A}$, where the current state is T are those of the cell 1(1) at times 3, 4 and 5, and those of cell 0 at times 5 and 6. Only one rule contains two consecutive occurrences of $\mathbf{N}$: the rule $T\ \mathbf{N}\ \mathbf{N}\ T\ \mathbf{B}\ \mathbf{B}\ T$. Note that the relative positions of these two occurrences of $\mathbf{N}$ and that of T are fixed. Now, when T occurs in a cell of the track, it is the single occurrence of this symbol on the track as this is the case with $\mathcal{L}$. In particular, in $(**)$, if $\alpha_0 = T$, then α_1 or α_{-1} must be $\sqcup$ and the other symbol is any one of $\mathcal{L}$ except the blank and T: see the table of the rules of $\mathcal{L}$ in [1]. Consequently, a rule of the track when the current state is T cannot be confused with the rules with T as the current state in the rules for $\mathcal{A}$.

Now, consider the case when $\mathbf{B}$ is 0 or A. We compare $(**)$ with the rules for $\mathcal{A}$ which have the same symbol as the current state. When it is A, the rules for $\mathcal{A}$ of $\mathbf{N}$ while there are only two of them in the rule $(**)$. For symbol 0, we have a similar

argument: the rules $(**)$ with 0 as the current state have at least three consecutive neighbours in $\mathbf{N}$.

We remain with the case when $\mathbf{B}$ is the blank of $\mathcal{L}$. From the previous cases, we know that there is no possible confusion when the rule contains at least three consecutive occurrences of $\mathbf{N}$. Now, two rules have two consecutive occurrences of $\mathbf{N}$ exactly: the rule $\mathbf{B}\ \mathbf{N}\ \mathbf{N}\ \mathbf{B}\ T\ A\ \mathbf{B}$ and the rule $\mathbf{B}\ \mathbf{N}\ \mathbf{N}\ \mathbf{B}\ T\ \mathbf{B}\ \mathbf{B}$. These rules can be seen as rules of the form $(**)$ when $\mathbf{B}=\sqcup$. The corresponding rules are $\sqcup\ \mathbf{N}\ \mathbf{N}\ \sqcup\ T\ A\ \sqcup$ and $\sqcup\ \mathbf{N}\ \mathbf{N}\ \sqcup\ T\ \sqcup\ \sqcup$, respectively. Now, interpreted as rules induced by a rule of $\mathcal{L}$, the corresponding rules of $\mathcal{L}$ would be $\sqcup\sqcup A \to \sqcup$ and $\sqcup\sqcup\sqcup \to \sqcup$. Now, these rules are indeed present in the table of the rules of $\mathcal{L}$, see [1]. And so, at this stage, there is no confusion by replacing $\mathbf{W}$, $\mathbf{W}_0$ and $\mathbf{B}_0$ by T, 0 and A respectively.

6.2. The erasing of the support of the track

We have to look at the final stage of the process. When the halting is met, $\mathbf{N}$ is introduced onto the track and, as we know from Section 5, this starts the erasing process of the support of the track by propagation of $\mathbf{N}$ which successively replaces all occurrences of T and, at the end of the propagation of the ray, which stops the production of cells in 0.

This requires the following rules:

0	T :	$\mathbf{N}\ \mathbf{N}\ T\ \mathbf{N}\ T\ \mathbf{N}$		T :	$\mathbf{N}\ \mathbf{N}\ \mathbf{W}_1\ \mathbf{W}_1\ \mathbf{B}\ \mathbf{N}$
0	$\mathbf{N}$:	$\mathbf{N}\ \mathbf{N}\ T\ \mathbf{N}\ T\ \mathbf{N}$		T :	$\mathbf{N}\ \mathbf{N}\ 0\ 0\ \mathbf{B}\ \mathbf{N}$
1(1)	T :	$\mathbf{N}\ \mathbf{N}\ \mathbf{N}\ \mathbf{B}\ T\ \mathbf{N}$		$\mathbf{N}$:	$\mathbf{N}\ \mathbf{N}\ \mathbf{N}\ \mathbf{B}\ T\ \mathbf{N}$
1(4)	T :	$\mathbf{N}\ \mathbf{N}\ \mathbf{N}\ T\ \mathbf{B}\ \mathbf{N}$		$\mathbf{N}$:	$\mathbf{N}\ \mathbf{N}\ \mathbf{N}\ T\ \mathbf{B}\ \mathbf{N}$

Now, it is easy to see that none of them cannot be confused with the rule $T\ \mathbf{N}\ \mathbf{N}\ T\ \mathbf{B}\ T\ T$ nor a rule of the form $(**)$ as these latter rules have only two occurrences of $\mathbf{N}$. It can also be seen that the above rules cannot be confused with any of the other rules for $\mathcal{A}$ where the current state is T: again the number of occurrences of $\mathbf{N}$ is different. Indeed, this number is two or three in the above rules while it is at most one only in the rules for $\mathcal{A}$ except the rules $T\ \mathbf{N}\ \mathbf{N}\ T\ \mathbf{B}\ \mathbf{B}\ T$ and $T\ \mathbf{N}\ \mathbf{N}\ T\ \mathbf{B}\ T\ T$. But there can be no confusion with these latter rules either: they have T after the two occurrences of $\mathbf{N}$ while in the above rules with two occurrences of $\mathbf{N}$ exactly, which are also consecutive, there is 0 or $\mathbf{W}_1$ after the second $\mathbf{N}$.

We also have to check that the track is not disturbed by the replacement of T by $\mathbf{N}$. Indeed, this replacement has, as a consequence, that the form $(**)$ is replaced by the following one:

$$\alpha_0\mathbf{N}\mathbf{N}\alpha_{-1}\mathbf{N}\alpha_1\alpha_0^1 \qquad (***)$$

As the three occurrences of $\mathbf{N}$ in $(***)$ are not consecutive, there is no confusion with the rules for $\mathcal{A}$ where the current state is $\mathbf{B}$.

This completes the proof that $\mathcal{C}$ exactly simulates the computation of $\mathcal{L}$ with a true stopping of the cellular automaton in the case when the computation of $\mathcal{L}$ also stops. Accordingly we have proved the following result:

Theorem 6.1. *There is a rotation invariant hyperbolic cellular automaton in the pentagrid which starts from finite configurations and whose halting problem is undecidable which has* 9 *states exactly, the blank included.*

7. Conclusion

While stating Theorem 4.1, we have explained why we did not say that the cellular automaton $\mathcal{A}$ is strongly universal and the same explanation holds for the automata $\mathcal{B}$ and $\mathcal{C}$ of Theorems 5.1 and 6.1 respectively.

Can we still have a strongly universal cellular automaton with 9 states or possibly less?

One way to solve this problem would be to apply the technique of [1] to another small Turing machine. In [11] where the defect of this machine was first noticed, the author provides another Turing machine with 7 states and 4 letters which mimics any tag system of a given family, the same as for Minsky's machine, and the machine of [11] is actually universal. Another Turing machine with 7 states and 4 letters which is truly universal was later provided by R. Robinson, see [10]. It would be interesting to see whether a smaller machine, as the one devised by T. Neary and D. Woods with 6 states and 4 letters, see [9], could yield a better solution.

Accordingly, there is some work ahead, probably a tedious one if not more difficult.

Acknowledgement

The author is very much in debt to the referees for their remarks allowing him to improve the paper.

References

[1] Lindgren K. and Nordahl M.G., Universal computation in simple one-dimensional cellular automata. Complex Systems, **4**, 299–318, (1990).

[2] M. Margenstern, New Tools for Cellular Automata of the Hyperbolic Plane, *Journal of Universal Computer Science*, **6**(12), (2000), 1226–1252.

[3] M. Margenstern, Implementing Cellular Automata on the Triangular Grids of the Hyperbolic Plane for New Simulation Tools, **ASTC'2003**, (2003).

[4] M. Margenstern, Cellular Automata in Hyperbolic Spaces, Volume 1, Theory, *OCP*, Philadelphia, (2007), 422p.

[5] M. Margenstern, Cellular Automata in Hyperbolic Spaces, Volume 2, Implementation and computations, *OCP*, Philadelphia, (2008), 360p.

[6] M. Margenstern, A universal cellular automaton on the heptagrid of the hyperbolic plane with four states, *Theoretical Computer Science*, (2010), accepted.

[7] M. Margenstern, About the embedding of one dimensional cellular automata into hyperbolic cellular automata, *arXiv*:1004.1830[cs.FL], (2010), 19pp.

[8] M. Margenstern, An upper bound on the number of states for a strongly universal hyperbolic cellular automaton on the pentagrid, *arXiv*:1006.3451[cs.FL], (2010), 17pp.

[9] T. Neary, D. Woods, Four Small Universal Turing Machines, *Fundamenta Informaticae*, **91**(1), (2009), 123-144.

[10] R. Robinson, M. Minsky's small universal Turing machine, *International Journal of Mathematics*, **2**(5), (1991), 551-562.

[11] Yu. V. Rogozhin, Sem' universal'nykh mashin T'juringa. *Matematicheskie Issledovanija*, **69**, 76-90, 1982 (Seven universal Turing machines) (in Russian)

This work is licensed under the Creative Commons Attribution-NoDerivs License. To view a copy of this license, visit `http://creativecommons.org/licenses/by-nd/3.0/`.