Complex Systems **8** (1994) 347–366

Inherent Generation of Fractals by Cellular Automata

Bruno Martin*
Laboratoire d'Informatique, Signaux et Systèmes de Sophia Antipolis,
I3S-U.R.A. 1376-C.N.R.S., Université de Nice-Sophia Antipolis,
650, route des Colles, B.P. 145,
06903 Sophia Antipolis Cedex, France

Abstract. In this paper we propose a method for generating fractal patterns using "classical" cellular automata. Although the problem of fractal generation for linear cellular automata has been studied recently, this is not the case for classical cellular automata.

We first exhibit some basic techniques for the construction of the transition function, which draws a Cantor set, and show how this method can be generalized to cellular spaces of greater dimension. Then we give a method for embedding the configurations into a closed interval to obtain fractal patterns. We also define discrete dynamical systems for counting the minimum number of balls required to cover the fractal pattern.

1. Introduction

Cellular automata (CAs) are of great interest for modeling complex physical systems or synchronous parallel processes. One point of interest is fractal generation. Indeed fractals, and more precisely Cantor-like sets, occur quite often as attractors for simple maps in physics. For instance, the Feigenbaum attractor has for a unique attractor a Cantor set [13]. Fractals also appear in sequences of errors in the transmission of data [8].

The problem of fractal production or fractal behavior of CAs during their long time evolution has been studied recently by Wolfram [23], Culik [4], Willson [21], and Haesler et al. [7]. These authors often deal with linear modulo-2 unidimensional CAs whose main instance is the Pascal triangle, which can be generated finitely by algebraic means [10]. Moreover, Willson gives a nice theorem about the Hausdorff dimension (or fractal dimension) but with a formalism quite difficult to apply that relies heavily on the linearity of the rules. His results have been improved and generalized to linear modulo-q CAs (see [22]).

*Electronic mail address: bmartin@unice.fr

Haesler et al. [7] claimed that the production of fractal structures in the long time evolution of CAs poses three major problems:

1. When and in what sense is there a limit set for the evolution of a CA?
2. How can the self similarity features of a limit set be formally modeled and deciphered?
3. Which classes of CAs generate fractal structures?

Willson [21] has discussed the first item and Takahashi [19] the second one. Haesler et al. considered all three problems and connected the deciphering operation to a matrix substitution system.

The goal of this paper is to give another formalism for the generation of fractals, based mostly on Cantor sets and products of Cantor sets (e.g., the Sierpinski carpet or Sierpinski-Menger sponge), and to give an example of the computation of the fractal dimension within the world of discrete mathematics. Moreover, we also give discrete dynamical systems associated with the configurations, which correspond to the Hausdorff measure.

We note that, unlike previous work, our formalism does not require linear local transition functions to compute the fractal dimension. It is thus easier to build a specific CA. We usually work with an embedding of some configurations into a continuous space, and can restrict ourselves to a subsequence of the configurations in the discrete space to make computations. Our formalism gives a good idea of the convergence of some discrete patterns to their continuous definition with the following idea: the radius of the covering balls for the pattern in the continuous space decreases as the length of the configurations in the discrete space increases. In other words, if a real quantity converges to zero, its discrete representation converges to infinity.

2. Definitions

2.1 Formal definition

Let us recall briefly the definition of a *fractal set*. According to Mandelbrot [8, 9], a set X is called a *fractal* provided its *Hausdorff dimension* $h(X)$ is not an integer. Intuitively, $h(X)$ measures the growth of the number of sets of diameter ε needed to cover X when $\varepsilon \to 0$. More precisely, if $X \subset \mathbf{R}^m$, let $N(\varepsilon)$ be the minimum number of m-dimensional balls of diameter ε needed to cover X. Then, if $N(\varepsilon)$ increases like $N(\varepsilon) \to \varepsilon^{-d}$ as $\varepsilon \to 0$, one says that X has Hausdorff dimension d.

We remark that this definition relies strongly on the definition of the Hausdorff dimension defined by means of the Hausdorff measure. A rigorous definition [15] for $h(X)$ proceeds as follows.

Definition 1. *Let X be a subset of a metric space and let $d > 0$. The* d-dimensional outer measure $m_d(X)$ *is obtained from*

$$\begin{cases} m_d(X,\varepsilon) = \inf\{\sum_{i\in I}(\mathrm{diam}S_i)^d\}, \\ m_d(X) = \lim_{\varepsilon\to 0} m_d(X,\varepsilon). \end{cases}$$

where the inf is over all finite coverings of X by sets S_i with diameter less than $\varepsilon > 0$.

Depending on the choice of d, $m_d(X)$ may be finite or infinite. Hausdorff showed in 1919 that there is a unique $d = d^*$ at which $m_d(X)$ changes from infinite to finite as d increases. This leads to the definition

$$h(X) = \sup\{d \in \mathbf{R}^+ : m_d(X) = \infty\}$$

The formal definition of the Hausdorff measure implies that it is very difficult to compute the fractal dimension of a set. We give another definition that is more useful in computing the Hausdorff dimension.

2.2 Fractal covering dimension

Given a subset X of a metric space and $\varepsilon > 0$, let $N(\varepsilon)$ be the minimum number of balls of diameter ε necessary to cover X. An alternative definition for $h(X)$, called a *fractal covering dimension* or *box counting dimension*, is the following (see [8]):

$$h(X) = \lim_{\varepsilon \to 0} \inf \frac{\log N(\varepsilon)}{\log(1/\varepsilon)}$$

Then $N(\varepsilon) \to \infty$ as $\varepsilon \to 0$. If $N(\varepsilon) \sim K/\varepsilon^d$, then $d = h(X)$ (see [5] for more details). Indeed, $m_\varepsilon^{d'}(X) = N(\varepsilon) \times \varepsilon^{d'} \sim K\varepsilon^{d-d'}$. Then, if $d' < d$ we get infinite dimension and if $d' > d$ we get finite dimension.

This process gives good results for the cases of fractals defined by a ground pattern and recursive definition such as Cantor sets, von Koch sets, or Sierpinski carpets. We will thus compute the fractal dimensions of the sets we will consider by means of the box-counting dimension and not by means of the formal definition of the Hausdorff dimension. For self-similar sets that satisfy the *open set condition* (see [6]) the box-counting dimension equals the Hausdorff dimension ([6], Theorem 9.3). The open set condition can be stated as follows: given a set of similarities $S_1, \ldots, S_k : \mathbf{R}^n \to \mathbf{R}^n$, where each S_i transforms subsets of $\mathbf{R}^n$ into geometrically similar sets, we say that the S_i satisfy the open set condition if there exists a non-empty bounded open set V such that

$$V \supset \bigcup_{i=1}^{m} S_i(V)$$

with the union disjoint. The fractal patterns we build satisfy this condition, and each "duplicating" process we define later can be understood as a particular similarity.

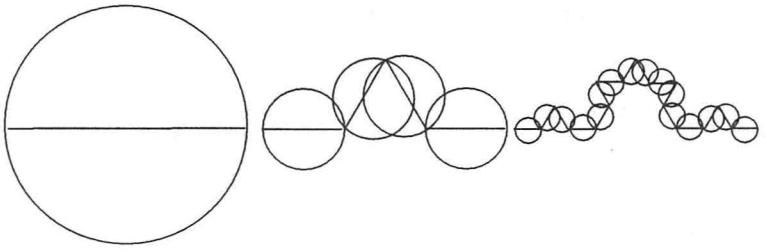

Figure 1: Coverings of the von Koch curve.

Example. We give the example of the computation of the fractal dimension of the von Koch curve defined by a ground pattern and recursive definition. Figure 1 depicts such a von Koch set where N denotes the minimum number of balls of diameter d necessary to cover the set.

The first part of the figure depicts the case where the radius of the ball equals one; the next two parts depict the increasing number of balls necessary as the diameter decreases by a factor of three. We then have the following equality:

$$\frac{K}{(\varepsilon/3)^d} = \frac{4 \times K}{\varepsilon^d}$$

Thus, we get $3^d = 4 \Rightarrow d = \log 4 / \log 3$, which is the well-known result for the fractal dimension of the von Koch curve. The number of balls drawn on the figure is exactly the minimum number of covering balls, which can be shown by means of the covering of the flipped V and the self-similarity of the curve.

In the next section we recall briefly some results of authors who have dealt with the relationships between *linear cellular automata* and fractals.

2.3 Cellular automata

In this section we present a constructive definition for cellular automata that is close to von Neumann's classical definition.

Definition 2. *A* cellular automaton *is a d-dimensional infinite array of identical cells indexed by* $\mathbf{Z}^d$. *Each cell is a finite state machine* $C = (Q, \delta)$ *where*

- Q *is a finite set, the set of the states*
- δ *is a mapping such that* $\delta : Q \times Q^\nu \rightarrow Q$

where ν *denotes the number of neighbors.*

An important notion in dealing with cellular automata is the notion of *configuration.*

Definition 3. *A* configuration *of an n-dimensional cellular automaton with a set of states denoted by Q is a mapping C of $\mathbf{Z}^n \to Q$, which assigns a state of Q to each cell (considered as a point of $\mathbf{Z}^n$) of the cellular automaton.*

Configurations are useful devices in representing the evolution of the cells in time.

The interesting aspect of the cellular automata is their temporal evolution, which can be regarded as a sequence of configurations. Such a *graph* is called a time-space diagram and is useful in designing algorithms for cellular automata. Although such a diagram is easy to draw in the plane, it is more difficult to consider for higher-dimension cellular spaces. In this case, the sequence of configurations can be drawn in a three-dimensional discrete space with two axes denoting the coordinates of a cell and the third one denoting the time. In the rest of this paper we will not give such representations in three dimensions, which are difficult to read; rather, we will consider their projections on the plane.

2.4 Linear cellular automata

We recall briefly the definition of a *linear cellular automaton* (LCA). We denote by $\mathcal{P}^n$ the set of all configurations of a Q-state's n-dimensional cellular automaton. Thus, $\mathcal{P}^n = Q^{\mathbf{Z}^n}$. We define a *global dynamics* G on $\mathcal{P}^n$ as follows.

Definition 4. *A* global transition function *G on the set of all the configurations $\mathcal{P}^n$ is a map $G : \mathcal{P}^n \to \mathcal{P}^n$ such that*

- *there exists a* quiescent state,
- *there exist m neighbors $(\nu_i)_{i=1,\ldots,m} \in \mathbf{Z}^n$ and a map $g : Q^m \to Q$ such that $\forall \nu \in \mathbf{Z}^n \ \forall \omega \in \mathcal{P}^n$,*

$$G(\omega(\nu)) = g(\omega(\nu + \nu_1), \ldots, \omega(\nu + \nu_m))$$

Then, the transition rule G on $\mathcal{P}^n$ is *linear* provided its *generating function* g is linear or, equivalently, provided

$$G(\omega + \tau) = G(\omega) + G(\tau).$$

Example. It is easy to see that a trivial cellular automaton given by the global dynamics $T(x)_i = 0$ is linear. Another example is $T(x)_i = x_{i-1} + x_i + x_{i+1} \bmod 2$ denoting the XOR cellular automaton of radius 1.

A linear cellular automaton can also be defined in the following way. We consider a dynamics $S_k : \mathcal{P}^n \to \mathcal{P}^n$ such that $S_k(x)_i = x_{i-k}$. Such a dynamics is called a *shift*. Its effect is to translate the input configuration x in the direction of k so that cell k assumes the state of cell 0. The definition of shift leads to the next definition.

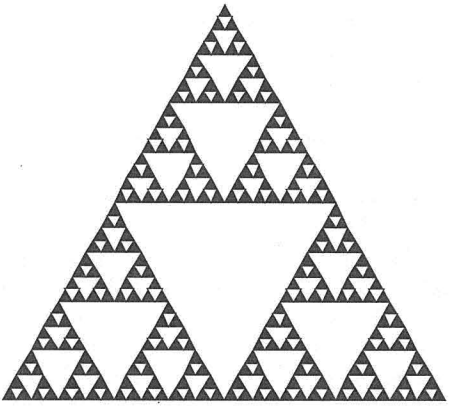

Figure 2: Pascal's triangle.

Definition 5. *If a linear global dynamics $T(a)$ satisfies*

- $T(0) = 0$
- *T commutes with shifts*
- *$T(a)$ has a finite support*

then T is a cellular automaton.

In this case, linearity is equivalent to the local condition $g(x_0, x_1, \ldots, x_m) = \sum a_i x_i$ for some elements from the set of states $a_i \in Q$.

We will prefer classical definitions for cellular automata because they do not necessarily require a linear form for the generation of fractals.

3. LCAs and fractals: some results

Culik et al. [4] have shown that the regular evolution of linear cellular automata on simple initial configurations generates a pattern that might be fractal or self-similar. The patterns they obtain are often similar to Pascal's triangle (see Figure 2). Their main result is the following.

Theorem 1. *Let f be the XOR cellular automaton rule for radius r. Then, for all n, where n is a power of 2,*

$$G_F^n(\omega) = \omega^{2r+1}$$

for every finite configuration of length n. Note that ω may start or end with zeros.

It is then extended to another type of cellular automata, namely Trellis automata. Reimen [16] proves a similar result with an algebraic property of superposability.

In many aspects, Willson [21] extends the results of Culik et al. He shows that if L denotes the global transition rule on cellular automata taken linearly modulo 2, a compact subset of Euclidean space related to the behavior of L is defined. This subspace can have fractional Hausdorff dimension. Willson has recently extended these results to linearly modulo q cellular automata [22].

This invariant object L somehow summarizes the infinite run of the global dynamics T and turns out to be independent of the initial configuration. This object, called *lim* T, is an invariant and fractal set. The computation of its Hausdorff dimension is made by the explicit construction of its coverings as summarized in the next theorem.

Theorem 2. *Let $X \subset \mathbf{R}^n$ be compact.*

1. *Suppose for some integer $a \geq 2$ and vectors $\nu_1, \ldots, \nu_m$ we have $aX \subseteq \bigcup_{i=1}^{m}(X + \nu_i)$. Then, gdim $X \leq \log_a m$.*
2. *Suppose for some integer $a \geq 2$ and vectors $\nu_1, \ldots, \nu_m$ that are pairwise distinct mod a we have $aX \supseteq \bigcup_{i=1}^{m}(X + \nu_i)$. Then, gdim $X \geq \log_a m$.*
3. *If $aX = \bigcup_{i=1}^{m}(X+\nu_i)$ where a and the ν_i are as in part 2, then gdim $X = \log_a m$.*

In this case, the topological similarity dimension is equal to the Hausdorff dimension. For a most detailed version of the result see [21]. But in this case, the fractal dimension is still very difficult to compute and the patterns generated strongly resemble the usual Pascal's triangle.

The last result we recall here gives a more convenient way to compute the fractal dimension. The result is from Takahashi [19].

Theorem 3. *The limit set of a p^k-state's LCA can be represented as the union of some members of a family of sets X_j that have the following property: each X_j is composed of n_{qj} $1/p$-scaled X_q's. If a transition matrix is defined by $A = (n_{qj})$, then the Hausdorff dimension of the limit set is given by $\log_p \lambda$, where λ is the maximum eigenvalue of the matrix A.*

More intuitively, Takahashi's idea is the following. The matrix consists of the count of subpatterns into a pattern, and taking the maximum eigenvalue counts (in a certain sense) the irregularities of the pattern. Then, the computation of the fractal dimension is quite simple.

4. Another way to generate fractals

We first give two simple examples of the generation of fractal patterns, one in dimension 1 and another in dimension 2. These two examples are good illustrations of the power of generating fractal patterns without using the linearity of the transition function. As we will see, it is possible to generate patterns like a Cantor set and the Sierpinski-Menger carpet, which appears to be novel in the cellular automata literature.

4.1 Some preliminary results

We present some definitions and results that will be useful later in the paper. Let us first give an "extended" definition for cellular automata in which we allow the set of the states to be the cartesian product of finitely many finite sets.

Definition 6. *We call a* tuple cellular automaton $D = (Q, \delta)$ *any cellular automaton with the set of the states equal to the cartesian product of finitely many sets, that is,* $Q = Q_1 \times \cdots \times Q_k$, *and with local transition function* $\delta : Q^\nu \times Q \to Q$ *such that* $a = \delta(x_1, x_2, \ldots, x_\nu, x)$ *for* x_i, $1 \leq i \leq \nu$ *neighbors of cell* x *and where* $a, x, x_i, 1 \leq i \leq \nu$ *are* k*-tuples.*

Observe that this definition is similar to the usual definition of cellular automata, that is, the transition function depends on all the elements of the tuples and is not the cartesian product of some transition function as is the definition of a cellular automaton product as defined in [1]. Furthermore, any tuple cellular automaton can be transformed into a "classical" automaton if we map any k-tuples onto a single element by means of a classical bijection from $N^k \to \mathbf{N}$, for instance. We will call a *layer* any element of the k-tuples (see Figure 3).

Notice that the above definition lets us define the transition function in terms of combinations of some simple transition functions. Below we will give some examples of simple transition functions, one that solves the firing squad synchronization problem and one that duplicates a finite configuration.

4.1.1 The firing squad synchronization problem

The *firing squad synchronization problem* is due to Myhill (1957) and can be expressed as follows.

> *Given an initial line of soldiers, how can they fire at the same time knowing that the order to fire, coming from a general located at one end of the line, needs a certain constant to propagate?*

Each soldier may be represented by one cell of a cellular automaton. The problem is then to build a local transition function for a cellular automaton. The first answer was given in 1965 by Minsky and MacCarthy. First minimal solutions are due to Goto, Waksman, and Balzer. Several years later, a minimal time solution with 6 states was given by Mazoyer [11].

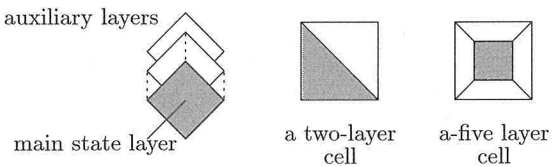

Figure 3: Some representations of tuple cells.

We will use the following results to synchronize segments of the automaton [20, 3].

Lemma 1 (Firing Squad Lemma) *There exists a CA $Z = (Q, \delta)$ with special symbols ç,\$ $\in Q$ and a quiescent state q such that*

$$\Delta^t({}^\omega q 0^{n-1} ç q^\omega) = {}^\omega q \$^n q^\omega$$

for $t = 2n - 2$ and $\Delta({}^\omega q 0^{n-1} ç q^\omega)(i,t) \neq \$$ for $0 \leq t < 2n - 2$, where Δ denotes the global transition function corresponding to the local transition function δ.

The states have the following meaning. The cell with the special symbol ç is the general who gives the order to fire. At the end of a certain process, all the "soldiers" are ready to fire (that is, the cells enter special state \$).

It is also possible to improve the time of synchronization if the initial line has not one general but two. In that case we have the following result [12].

Lemma 2 (Firing Squad Lemma with two generals) *There exists a CA $Z = (Q, \delta)$ with special symbols ç,\$ $\in Q$ and a quiescent state q such that*

$$\Delta^t({}^\omega q ç 0^{n-2} ç q^\omega) = {}^\omega q \$^n q^\omega$$

for $t = n$ and $\Delta({}^\omega q ç 0^{n-2} ç q^\omega)(i,t) \neq \$$ for $0 \leq t < n$, where Δ denotes the global transition function corresponding to the local transition function δ.

The two ç's on the initial configuration denote the two generals. This is an immediate consequence of Lemma 1.

For the proofs of the previous lemmas, refer to [3, 12, 14, 20].

4.1.2 Duplicating a finite configuration

Many cellular automata designers might have noticed the usefulness of a local transition function that moves or duplicates a finite configuration to the right or to the left. We aim to describe such a local transition function in this section because it will be our building block for the transition functions we will describe for generating fractal patterns.

We are faced with the following problem.

> *Given an initial configuration of the form $x = x_1 \dots x_n$ with $x_i \in Q$ for $1 \leq i \leq n$—in other words x is a finite word over the set of the states of the cellular automaton—we want to obtain a configuration given by $xx = x_1 \dots x_n x_1 \dots x_n$.*

This problem can be solved by the following process, illustrated in Figure 4. At an initial time, the first cell sends its main state x_1 on the auxiliary layer. This message moves as quick as possible to the right until the main layer of the right neighbor of the cell that contains the "flying" state x_1 is a quiescent

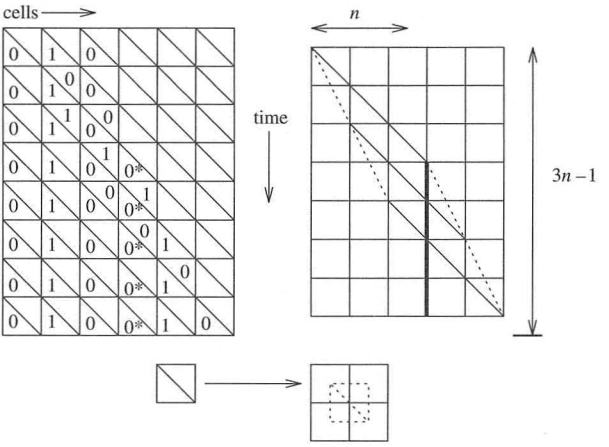

Figure 4: Duplicating a finite configuration: time-space states diagram and time-space diagram.

state. We thus have the following transition, where the couples have to be interpreted as main layer × auxiliary layer:

$$((x_n, x_1), (q, q), (q, q)) \to (x_1^\star, q) \tag{1}$$

where $x_1^\star$ serves as an "end of configuration" delimiter. It remains to define how the rest of the configuration can be sent. We first have to define how a cell can be ready to emit. When x_1, the first state sent, moves to the right, it switches $x_2 \ldots x_n$ to the "ready" state, for instance $\underline{x}_2, \ldots, \underline{x}_n$:

$$((\underline{x}_{i-1}, x_2), (x_i, x_1), (x_{i+1}, q)) \to (\underline{x}_i, x_2) \tag{2}$$

The rule for emitting the other main states is quite simple: they are sent as soon as possible when they are in a "ready" state. Consider, for instance, the emission of cell i already in a "ready" state. x_i is emitted when the auxiliary layer of its left neighbor is in a quiescent state and when the auxiliary state of cell i contains x_{i-1}. In fact, cell i does not need to know if its auxiliary state contains x_{i-1} because x_{i-1} will be the last state emitted before x_i can be emitted. We then have the following rule:

$$((x_{i-1}, q), (\underline{x}_i, x_{i-1}), (\underline{x}_{i+1}, x_{i-2})) \to (x_i, x_i) \tag{3}$$

which also turns $\underline{x}_i$ back to x_i.

The "flying" states stop when they encounter a situation similar to the one given by rule (1):

$$((x_{i-1}, x_i), (q, q), (q, q)) \to (x_i, q) \tag{4}$$

The time required for the duplication of an n-cell configuration is $3n-1$: the contents of the cells require $n+1$ time units to be placed in their right places,

and states are emitted every two time steps according to the definition of the above transition function. Thus, the contents of cell n is emitted at time $2(n-1)$ and moves $n+1$ units of time.

We have thus the following lemma.

Lemma 3. *There exists a cellular automaton $D = (Q, \delta)$ that duplicates an n-cell configuration in $3n-1$ time units. Moreover, if the configuration is given by a word $x = x_1 \ldots x_n$ over alphabet A, the set of states Q of the duplicating cellular automaton is $Q = (A \cup \underline{A} \cup A^*) \times A$, where $\underline{A}$ stands for the "ready" states and A^* for all possible "end of configuration" delimiters.*

4.2 A unidimensional CA that generates a Cantor set

We define the behavior of a one-dimensional cellular automaton that embeds a Cantor set in the closed interval $H = [0,1]$ of $\mathbf{R}$, also called a *Hilbert cube*. Its definition is quite clear and proceeds as follows.

Let $\mathcal{A} = (Q, \delta)$ be a one-dimensional cellular automaton with states $Q = \{0, 1, q\}$ (where q denotes the quiescent state) and some other auxiliary states. Let δ be the local transition function (or generating function) for the cellular automaton. In the following we will not give all the details of the transition function because the transition rules of the cellular automaton are very complicated. An explicit presentation of it would be unwieldy and nontransparent.

The process consists of steps starting from the initial configuration 010, or equivalently from the "white-black-white" configuration. The steps follow and are depicted in Figure 5.

1. Copy the configuration once to the right.
2. Copy the configuration a second time at the right end and paint the first copy in black.
3. Update the configuration and return to step 1.

Clearly, such a cellular automaton may be constructed. The duplication of a finite configuration can be done using the transition function described in section 4.1.2.

Hence, by iterating that process and embedding the restriction to $\{0,1\}$ of a subsequence of the configurations, we get the geometrical construction of a Cantor set. The total time between two "embeddable" configurations is thus $(7 \times$ length(finite configuration)$) - 2$ plus the time to come back, which is length(new finite configuration). These configurations occur at times given by the exponential sequence $1, 28, 278, \ldots$, or

$$\begin{cases} t_0 = 1 \\ t_{n+1} = 10t_n - 2 \end{cases}$$

The "drawing" of the Cantor set is then obtained by replacing in the subsequence all the occurrences of "0" by a segment and all the occurences of "1" by a hole, then resizing the configurations in [0,1]. The Cantor set is then the set of sites with symbol "0".

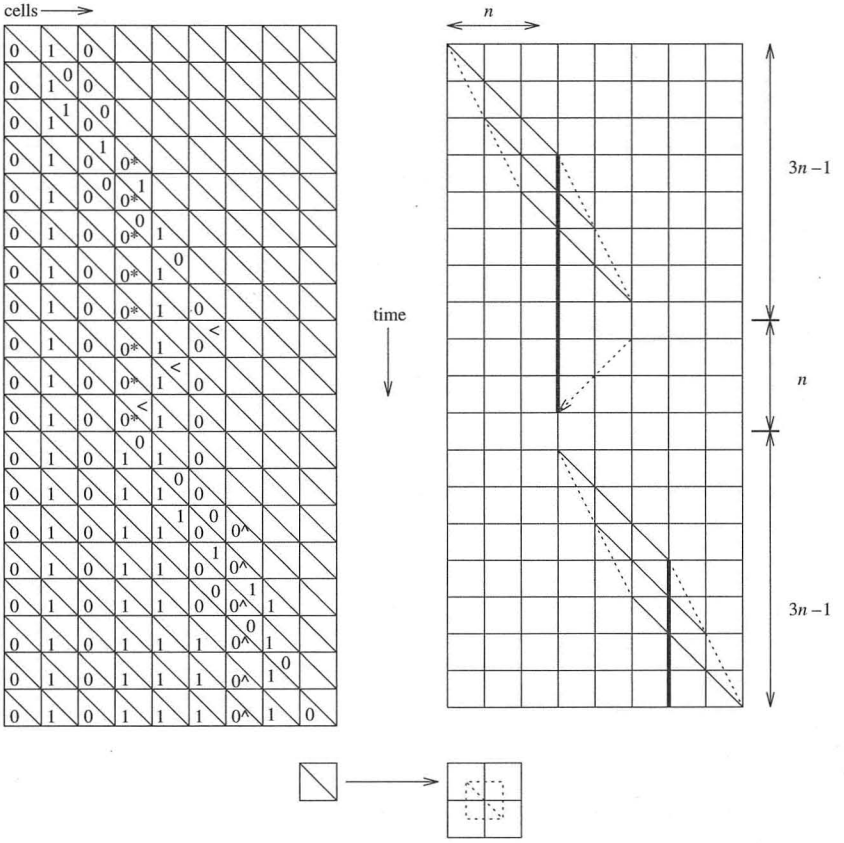

Figure 5: The two steps of the behavior: time-space states diagram and time-space diagram.

4.3 Generalization to cellular spaces of higher dimension

The process defined above can be generalized to higher-dimensional cellular spaces. In order to do that we first notice that the unidimensional cellular automaton described in the previous section can be modified as follows: instead of twice copying the initial configuration at the right end, it is also possible to copy it once at the left end and once at the right end. In other words, use the neighborhood vector of the cellular automaton to identify the part that is copied and has to be painted in black. With that modification, the cellular automaton expands to the left and to the right, without changing the continuous representation of the configuration. The point of interest of this definition is that the copies are made by using the neighborhood vector and might be generalized by means of this strategy. In this section, we describe the role of the neighborhood vector in a similar process.

4.3.1 Case of the von Neumann neighborhood

If we apply the process to the case of a two-dimensional cellular automaton with the von Neumann neighborhood, we get a direct product of two Cantor sets, one drawn horizontally and the other vertically. The process described in the previous section can easily be generalized from the one-dimensional case to higher dimensions according to the following observation: the two-dimensional cellular automaton acts on a line as a unidimensional one would act on a cell provided the line has already been synchronized by means of a firing squad process, such as the one with 6 states described in [11]. The case of the von Neumann neighborhood is not the most interesting one. In the next section we focus on the Moore neighborhood.

4.3.2 Case of the Moore neighborhood

The technique using the Moore neighborhood is the same as that using the von Neumann neighborhood. It suffices to synchronize a line before copying (see Figure 6). Here we will synchronize the left edge of the square. The synchronizing process then allows the synchronized line of cells to behave as if it were a unique cell in a unidimensional cellular automaton. Then, we apply the process depicted in Figure 5, which twice copies the initial finite configuration; in this case, however, we do not paint black (that is, rename the remaining 0s of the middle third in 1s) the middle third sub-configuration. In the case of a tuple cellular automaton that, for instance, emits the values of a line, it does not matter whether the crossing lines are synchronized. In the case of our process, we just have to obtain the first quiescent line and stop as soon as it is encountered, then copy one line into one of the directions

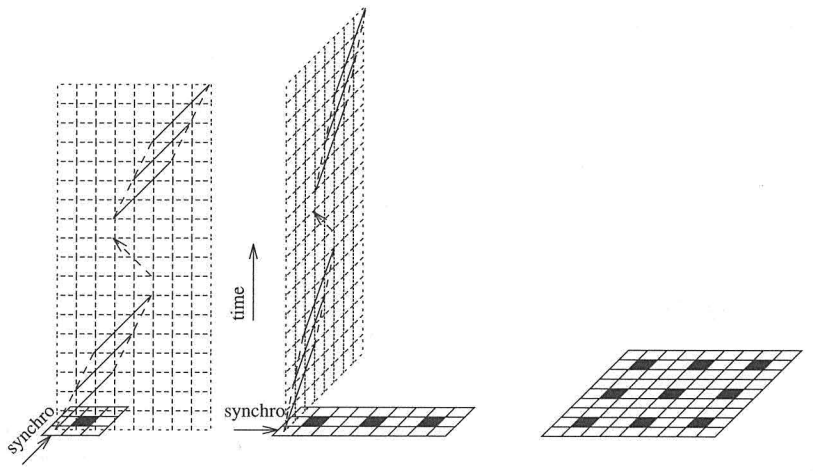

Figure 6: Copying the squares nine times: time-space diagram.

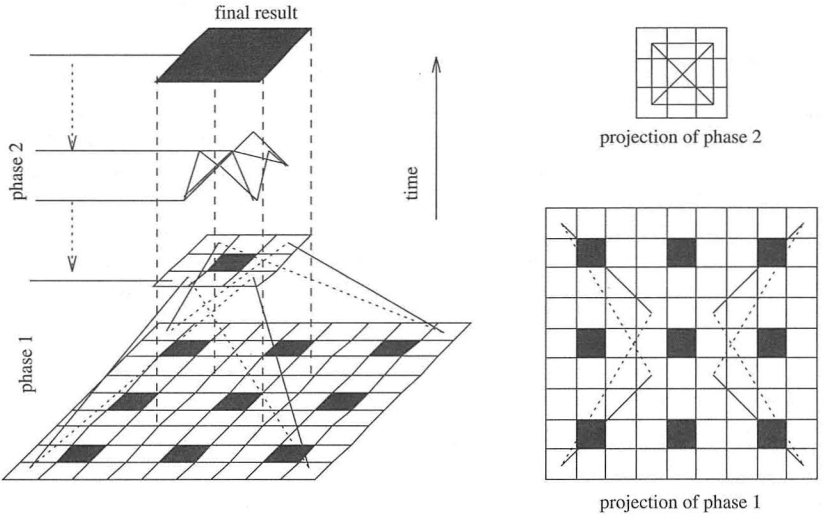

Figure 7: The signals emitted for identifying the middle third.

of the neighborhood. The return signal can be sent only (to the general) for the synchronization process.

We then synchronize again in the other direction—that is, on the largest square—and likewise copy twice the configuration with the layer containing the square to be painted black.

Finally, when the copying process is over, the corners send two signals, one of slope 1 at half speed and one of slope $\frac{1}{2}$ at full speed, in order to choose which square has to be painted black. The black color of the layer then enters the main state and paints black the middle third sub-configuration. Figure 7 depicts this action.

We summarize the process described above:

1. synchronize the configuration over the x axis;
2. apply the copying process twice along the y axis;
3. synchronize along the y axis;
4. apply the copying process twice along the x axis;
5. identify the four corners of the new configuration;
6. identify the middle third sub-square and "paint it black."

In the case of the Moore neighborhood, we get the promised Sierpinski carpet when embedding the configurations at exponential units of time. In the next section we give a more precise meaning to the assumption that we obtain something whose embedding maps to a continuous figure.

One can also easily see that such a process can easily be generalized to any neighborhood vector, giving other curious figures. Some of them may be fractals and others may not.

5. From configurations to fractals

In the previous section we presented the idea of generation. We need to make more precise which notion of covering we take in the continuous space to get an intuitive covering on the cells of the configuration, of an arbitrary dimension.

In section 4 we introduced a covering of black points of a proper subset of $\mathbf{Z}^n$. That is, we had to remove the covering of the intervals. In fact, the number of balls that interest us is the number of white points (or zero sites). In this case the covering is easier to describe. Indeed, if we are interested in the covering of the white points of the configuration, we need only cover the white points in the continuous interval, that is, the points that remain in the decomposition of the Cantor set. Thus, we cover the set by balls with the center being the white points and the diameter being a homothetical factor equal to 1/length(non-quiescent part of the cellular space).

In order to generalize the results, we take as balls the balls defined by the supremum norm, which gives balls with square form, or in higher dimensions with cubic form.

5.1 Usual Cantor set

In this section we describe how to cover the process defined for a one-dimensional cellular automaton with an embedding giving the usual Cantor set. The covering of the initial configuration gives, in the $H = [0, 1]$ interval of $\mathbf{R}$, the number of $N(\varepsilon) = 2$ with diameter $\varepsilon = 3^{-1}$ and in the configuration two white points with a homothetical ratio of $\frac{1}{3}$ that corresponds exactly to what we get in the continuous world and starts an induction. For the induction step, assume we have covered the nth configuration with 2^n balls. Clearly the length of the $n + 1$ configuration is 3^{n+1}, which is covered by 2×2^n balls (by the duplication), which is exactly 2^{n+1}. The rest of the configuration is painted black. We thus get the following result.

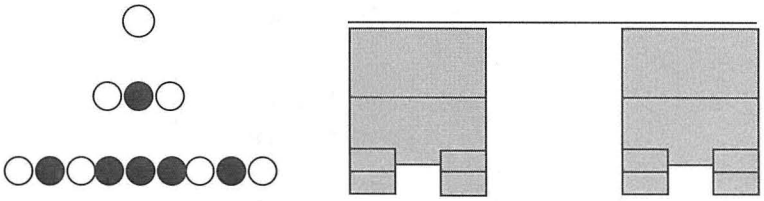

Figure 8: Comparison between the covering of the Cantor set and the number of white points.

Theorem 4. *The number of white cells of the configuration of the cellular automaton corresponds exactly to the size of the minimal covering of the Cantor set for balls of a diameter that corresponds to the homothetical factor. Furthermore, the Hausdorff dimension of the Cantor set embedding is* $\log 2/\log 3$.

Proof. Because $N_n(\varepsilon) = 2^n$ and $\varepsilon_n = 3^{-n}$, we have the equality $2^n = (3^{-n})^d$, which gives the dimension d given in the theorem. ∎

Furthermore, we can associate a discrete dynamical system that counts the proportion of black points on the configuration and, complementarily, one that counts the ratio of white points. The discrete dynamical system associated to the black points is the following. Let $\Pi_n^\bullet$ denote the proportion of black points. Clearly $1-\Pi_n^\bullet$ denotes the proportion of white points because of the quotient. The dynamical system can be defined recursively by

$$\Pi_n^\bullet = \frac{2}{3}\Pi_{n-1}^\bullet + \frac{1}{3}$$

which has 1 as an attractor. Thus $\Pi_n = 1 - \Pi_n^\bullet$ has 0 as an attractor, which maps to the definition of the Hausdorff measure of a Cantor set (see [5]).

We can now claim that the embedding of the subsequence of the sequence of the configurations of the cellular automaton defined previously converges to the usual Cantor set as a limit set.

5.2 Sierpinski Carpet

Here we give justification for the computation of the Hausdorff dimension in discrete space for the two-dimensional cellular automaton that computes the Sierpinski carpet depicted in Figure 9. The process for drawing the Sierpinski carpet on the mesh is the one suggested in section 4.2.2. Here we give a theorem analogous to Theorem 4.

Theorem 5. *The number of white points of the configuration of the 2-cellular automaton with the Moore neighborhood corresponds exactly to the*

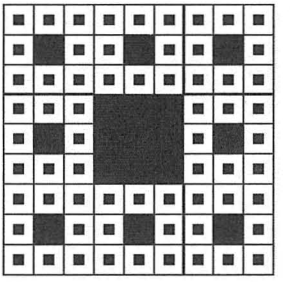

Figure 9: The Sierpinski carpet.

minimal covering of the Cantor set for balls of a diameter that corresponds to the homothetical factor. Furthermore, the Hausdorff dimension of its embedding is that of a Sierpinski carpet, namely $\log 8/\log 3$.

Proof. The covering of the initial configuration of the cellular automaton gives in $H^2 = [0,1] \times [0,1]$ a number of balls $N(\varepsilon) = 8$ with diameter $\varepsilon = 3^{-1}$, and in the configuration 8 white points with an homothetical ratio of $1/3$, which corresponds to the situation in H^2 and starts the induction. For the induction step, assume we have covered the nth configuration with 8^n balls having the same number of white points in the configuration. The area of the $(n+1)$th configuration is clearly $(3^{n+1})^2$, and contains (because of the duplications) 8×8^n white points with the same number of balls in its embedding, that is, 8^{n+1} balls. The rest of the nonquiescent part of the configuration is painted black.

Because $N_n(\varepsilon) = 8^n$ and $\varepsilon_n = 3^{-n}$, we get the equality $8^n = (3^{-n})^d$, which gives the dimension d of the theorem. ∎

As for the Cantor set, it is also possible to associate a discrete dynamical system with the Sierpinski carpet that counts the proportion of black points on the configuration and, complementarily, one that counts the proportion of white points and thus the proportion of balls. The discrete dynamical system associated with the black points can be determined as follows. Let $\Pi_n^\bullet$ denote the proportion of black points. Clearly $1 - \Pi_n^\bullet$ denotes the proportion of white points. The discrete dynamical system can be defined recursively by

$$\Pi_n^\bullet = \frac{8}{9}\Pi_{n-1}^\bullet + \frac{1}{9}$$

which also has 1 as an attractor. Thus $\Pi_n = 1 - \Pi_n^\bullet$ has 0 as an attractor, which maps to the definition of the Hausdorff measure of a Sierpinski carpet.

As for the Cantor set, we can claim that the embedding of the subsequence of the sequence of the configurations of the cellular automaton defined previously converges to the usual Sierpinski carpet as a limit set.

5.3 Sierpinski-Menger sponge

Even for higher dimensions, the process is analogous and leads to the same type of results. If we are interested in a three-dimensional cellular automaton with the generalization of the Moore neighborhood, we obtain the well-known Sierpinski-Menger sponge depicted in Figure 10 (see [8]). The usual theorem holds for three-dimensional cellular automata.

Theorem 6. *The number of white points of the configuration of the cellular automaton corresponds exactly to the minimal covering of the Sierpinski-Menger sponge for balls that correspond to the homothetical factor. Furthermore, the Hausdorff dimension of its embedding is that of the Sierpinski-Menger sponge,* $\log 26/\log 3$.

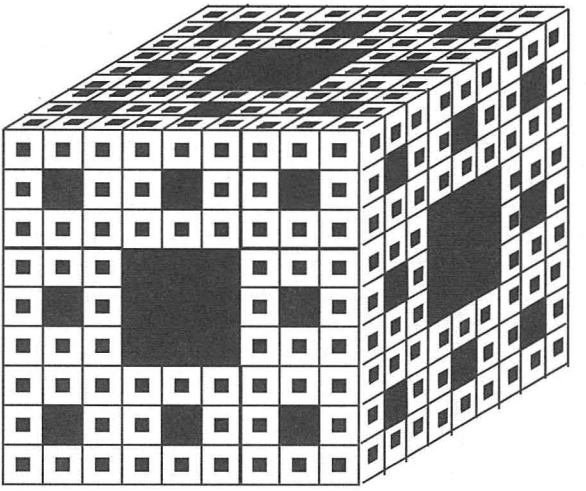

Figure 10: The Sierpinski-Menger sponge.

One can also determine the dynamical system that counts the proportion of black points of the nonquiescent part of the configuration, which is $\Pi^{\bullet}_n = \frac{26}{27}\Pi^{\bullet}_{n-1} + \frac{1}{27}$. The proportion of white points is denoted by $\Pi_n = 1 - \Pi^{\bullet}_n$.

Conclusion

Our method of generating fractals yields forms other than those obtained using linear cellular automata. Furthermore, the discrete dynamical systems we have defined seem to map to the definition of the Hausdorff measure. These examples clearly give instances of parallel exponential algorithms because their time-complexities is linear in the space-complexities, which grow exponentially.

There are many other methods for generating fractals, such as fractals obtained by $k \times k$ substitutions [2] and in relation to nice arithmetic properties listed in [17]. In [18], two additional methods for generating fractals are proposed: the first uses iterated Kronecker products, and the second uses iterated matrix-valued homomorphisms. Both provide efficient parallel algorithms for computing $n \times n$ images with $O(\log n)$ operations per pixel.

Acknowledgments

This work was supported by the Programme de Recherches Coordonnées du CNRS "*Mathématiques et Informatique*".

References

[1] J. Albert and K. Culik II, "A Simple Universal Cellular Automaton and Its One-Way and Totalistic Version," *Complex Systems*, **1** (1987) 1–16.

[2] J. P. Allouche, "Finite Automata in 1-D and 2-D Physics," pages 177–184 in *Number Theory and Physics*, edited by J. M. Luck, P. Moussa, and M. Waldschmidt (Berlin: Springer Verlag, 1990).

[3] R. Balzer, "An 8 State Minimal Time Solution to the Firing Squad Synchronization Problem," *Information and Control*, **10** (1967) 22–42.

[4] K. Culik and S. Dube, "Fractal and Recurrent Behavior of Cellular Automata," *Complex Systems*, **3** (1989) 253–267.

[5] K. J. Falconer. *The Geometry of Fractals Sets* (Cambridge: Cambridge University Press, 1985).

[6] K. J. Falconer, *Fractal Geometry*, 2nd edition (New York: Wiley, 1990).

[7] F. v. Haesler, H. O. Peitgen, and G. Skordev, "Cellular Automata, Matrix Substitutions and Fractals," *Annals of Mathematics and Artficial Intelligence*, **8** (1993) 345–362.

[8] B. B. Mandelbrot, *Les Objets Fractals* (Paris: Flammarion, 1975).

[9] B. B. Mandelbrot, *Fractals: Form, Chance and Dimension* (San Francisco: Freeman, 1977).

[10] O. Martin, A. M. Odlyzko, and S. Wolfram, "Algebraic Properties of Cellular Automata," *Communications in Mathematical Physics*, **93** (1984) 219–258.

[11] J. Mazoyer, "A Six State Minimal Time Solution to the Firing Squad Synchronization Problem," *Theoretical Computer Science*, **50** (1987) 183–238.

[12] J. Mazoyer and N. Reimen, "A Linear Speed-up Theorem for Cellular Automata," Research Report 91.16, Laboratoire d'Informatique Théorique et Programmation (1991). To appear in *Theoretical Computer Science*.

[13] J. Milnor, "On the Concept of Attractor," *Communications in Mathematical Physics*, **99** (1985) 177–195.

[14] E. F. Moore. *Sequential Machines, Selected Papers* (Reading, MA: Addison Wesley, 1964).

[15] H. O. Peitgen and P. H. Richter, *The Beauty of Fractals* (Berlin: Springer Verlag, 1986).

[16] N. Reimen, "Superposable Trellis Automata," pages 472–482 in *Mathematical Foundations of Computer Science*, volume 629 of *Lecture Notes in Computer Science*, edited by I. M. Havel and V. Koubek (Berlin: Springer Verlag, 1992).

[17] O. Salon, *Propriétés Arithmétiques des Automates Multidimensionnels*," PhD Thesis, Université de Bordeaux I (1989).

[18] J. Shallit and J. Stolfi, "Two Methods for Generating Fractals," *Computers and Graphics*, **13**(2) (1989) 185–191.

[19] S. Takahashi, "Cellular Automata and Multifractals: Dimension Spectra of Linear Cellular Automata," *Physica D*, **45** (1990) 36–48.

[20] A. Waksman, "An Optimum Solution to the Firing Squad Synchronization Problem," *Information and Control*, **9** (1966) 66–78.

[21] S. J. Willson, "Cellular Automata Can Generate Fractals," *Discrete Applied Mathematics*, **8** (1984) 91–99.

[22] S. J. Willson, "Calculating Growth Rates and Moments for Additive Cellular Automata," *Discrete Applied Mathematics*, **35** (1992) 47–65.

[23] S. Wolfram, "Universality and Complexity in Cellular Automata," *Physica D*, **10** (1984) 1–35.