

# The GraftalLace Cellular Automaton

Kaszanyitzky, András  
kaszi75@gmail.com

**Abstract.** We introduce our *GraftalLace Cellular Automaton* in short GLCA which is a new one-dimensional cellular automaton on the regular square lattice. It makes a *monochromatic infinite directed graph* which evolve deterministically row by row, by a defined rule and a single initial row of arc patterns. Arcs overlap each other and partly influence the states of the next cells in another arrangement. The data structure of GLCA is a *number triangle* or number trapezoid consists of octal digits formed by bit operations. We show examples of GLCA patterns which represent all four classes of Wolfram's classification. Some of the patterns belongs to *Sierpiński-like fractals* as *Pascal Triangle modulo 2* and *modulo 3* patterns which can be realized by GLCA in many different ways. We show these fractals, observe the *reversibility* of the rules and give ideas to extend our automaton by using more colours and other representations to find new interesting patterns.

## 1 Definition of GLCA

I invented GLCA in 1991 inspired by the articles of *Scientific American magazine* about elementary cellular automata of *Stephen Wolfram* [W84,W02] and graftal trees otherwise recursive fractal plants of *Aristid Lindenmayer* [D86,PL90]. Graftal is a combination of two words: graph + fractal.

GLCA connects root patterns with branch patterns through a junction (otherwise a grid point of the square lattice) by a defined rule. Both patterns are triplets of arcs formed by the incoming and the outgoing arcs of a junction from and into the same horizontal position with the left and right neighbour cells. Usually we get a porous, lace-like chaotic pattern. For illustration see *Figure 1*.

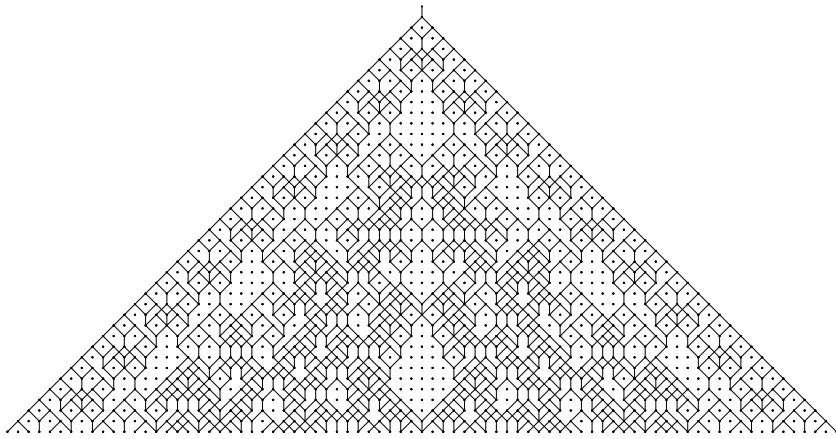
### 1.1 Formal definition of GLCA

GLCA operating with a chain of deterministic finite automata (DFA) and can be represented as a 4-tuple  $\langle \Sigma, \phi, B, c_0 \rangle$ , where  $\Sigma$  is an octal alphabet (cell states),  $\phi$  is the local transition function,  $B$  is a function to define the cell neighbourhood with bit operations and  $c_0$  is the initial configuration. GLCA evolves on an array of cells  $(s_l)$  where  $l \in \mathbb{N}$  and each cell takes a state from the octal alphabet. This

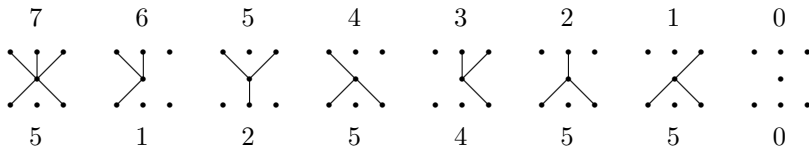
array represents a global configuration  $c$ , such that  $c \in \Sigma^*$ . The set of finite configurations of length  $l$  is represented as  $\Sigma^l$ . Cell states in a configuration  $c(j)$ , where  $j \in \mathbb{Z}$ , are updated by the next configuration  $c(j + 1)$  simultaneously by the local transition function ( $\phi$ ) otherwise the rule ( $R$ ). This rule tells how to transform all possible octal digits into another octal digit.

In the next subsections we show two bit operating functions:  $B_p(n)$  and  $B^q(m)$  to define the neighbourhood of the cells because transition function in GLCA unlike other cellular automata only partly influences the states of the next cells (3 cells, one at the same horizontal position with the left and right neighbour cells) and new states come from another arrangement of the new bit triplets.

The array increases maximum 2 cells in each time steps ( $j$ ). Evolution of GLCA is represented by a sequence of finite configurations ( $c_l$ ) given by the global mapping,  $\Phi : \Sigma^l \rightarrow \Sigma^{l+2}$ .



**Figure 1.** First 40 rows of Rule 51254550<sub>8</sub> with gridpoints, from a single vertical root. It was the first interesting GLCA pattern I have found in 1991.



**Figure 2.** A rule as an octal number  $R = 51254550_8$  means how to connect all the root patterns (upper triplets) with branch patterns (lower triplets in reverse order binary code) through the junction (centre point).

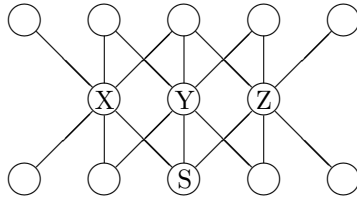
## 1.2 Basic definitions

Our *cell space* is the regular simple upright square lattice. All grid points are cells called the *junctions*. We use the Cartesian coordinate system with upside-down

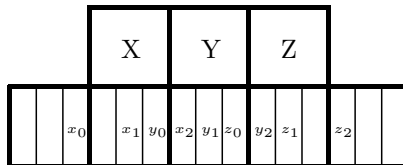
y-coordinates. We only allow connections between a cell and its closest 3 neighbour cells in a positive (nonzero) vertical direction with arcs and we denote the connection between two cells with a binary digit (1=connected, 0=independent). It means only vertical and diagonal arcs are allowed and the maximum number of the connecting arcs in one junction is six (3 indegrees and 3 outdegrees). We call the possible incoming arrangement of arcs into a junction: the *root pattern*. We call the possible outgoing arrangement of arcs from a junction: the *branch pattern*. We draw only the root pattern (states of the cells) in each time step and upload the next row with bits of the branches by the rule. Branches automatically become roots in another arrangement in the next time step.

Both patterns form binary triplets which have 8 possible variants denoted by an octal digit therefore we use the octal *alphabet*:  $\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7\}$ . The *state* of a cell ( $s$ ) specified by its incoming arc triplet, the root pattern. By choosing an eight-digits long octal number ( $R$ ) we get a *rule* for our GLCA which tells how to combine the potential root patterns with branch patterns. This rule gives the *local transition function* of GLCA:  $\phi(s)$ . Root patterns are denoted by the place values of the rule, their connecting branch patterns are denoted by the digits of the rule. See *mini trees* on *Figure 2* and more details on *Figure 5*.

For practical reasons we denote the branch patterns with a *reverse order binary number* because in the next time step (next row of the evolving pattern) branch arcs become root arcs in another arrangement where every arc belongs to different junctions in a reverse order. Both patterns overlap each other. *Figure 3* shows the overlapping patterns and the potential adjacency of neighbour cells.



**Figure 3.** Overlapping roots and branches. We show 3 cells in the middle row (X,Y,Z) with all of their possible connections. Their branches make a new root pattern by the incoming arcs of cell S.



**Figure 4.** Data representation of overlapping branch patterns from 3 junctions (X,Y,Z above) into 5 next junctions (below). Binary digits in a new combination represent a new root pattern:  $S = \overline{x_2y_1z_0}$ .

The rule defines the corresponding *branch pattern* for any potential root patterns otherwise for any potential states of the cells:  $s_{i,j} \rightarrow \phi(s_{i,j})$  which means 3 possible outgoing connecting arcs from the junction. These arcs form an octal digit as a reverse order binary triplet by the highest bit:  $(v_{i,j} \rightarrow v_{i+1,j+1})$ , the middle bit:  $(v_{i,j} \rightarrow v_{i,j+1})$  and the lowest bit:  $(v_{i,j} \rightarrow v_{i-1,j+1})$  where  $v$  is a cell otherwise a grid point (vertex). The horizontal position of the vertex in a row denoted by  $i$ , its vertical position otherwise the actual time step is denoted by  $j$  where  $i, j \in \mathbb{Z}$ .

One branch pattern (outgoing triplet of arcs from a junction) partly influences the states of its 3 different neighbour cells in the next row by changing their corresponding bits. The states of the cells come from their *root pattern* as their incoming connecting arcs from 3 different cells into a junction by the highest bit:  $(v_{i-1,j} \rightarrow v_{i,j+1})$ , the middle bit:  $(v_{i,j} \rightarrow v_{i,j+1})$  and the lowest bit:  $(v_{i+1,j} \rightarrow v_{i,j+1})$ .

Our rule is assigning all possible  $s$  values into not necessarily different  $\phi(s)$  values. The total number of the possible rules are  $8^8=16777216$ . We avoid growing branches from nothing therefore the last digit of the rule is always equal to zero. The number of the remaining rules is  $8^7=2097152$ .

### 1.3 Bit operating functions

In this section we show how the local transition function creates the states of the new cells in the next time step automatically (1). We define a new bit operating function  $B_p(n)$  which gives back the value of the  $2^p$  component of an octal digit  $n$ .

For example:  $B_2(6) = 4$ ,  $B_1(6) = 2$ ,  $B_0(6) = 0$ .

$$\phi : s_{i,j+1} = \sum_{p=0}^2 B_p(\phi(s_{i+1-p,j})) \quad (1)$$

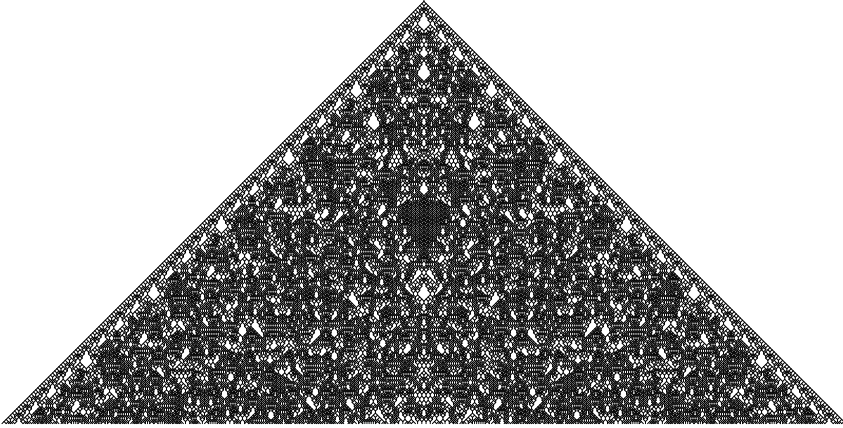
Let's consider another bit operating function  $B^q(m) = b$  which means let the  $q$ th bit of the octal number  $m$  is equal to bit  $b$ . For example: if  $m = 0$  then  $B^2(m) = 1$  means  $m = 4$ , and after that  $B^0(m) = 1$  means  $m = 5$ . Now we can show how the local transition function creates the branches (2) at the same time with root patterns. It is only another grouping of the arcs.

The following branches partly influence the states of 3 different cells in the next row:

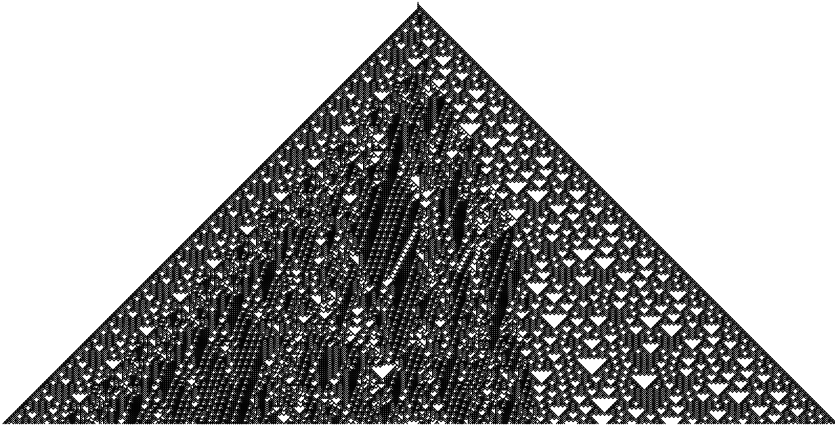
$$s_{i,j} \rightarrow \phi(s_{i,j}) = \begin{cases} B^2(s_{i+1,j+1}) = B_2(\phi(s_{i,j})) \\ B^1(s_{i,j+1}) = B_1(\phi(s_{i,j})) \\ B^0(s_{i-1,j+1}) = B_0(\phi(s_{i,j})) \end{cases} \quad (2)$$

See *Figure 4* for data representation of a new root pattern (new state of the cell S below cell Y) made by combined bits of different branch patterns. Branch

patterns with reverse order bits:  $X \rightarrow \overline{x_2x_1x_0}$ ,  $Y \rightarrow \overline{y_2y_1y_0}$ ,  $Z \rightarrow \overline{z_2z_1z_0}$  automatically make a new root pattern in the right order:  $S = \overline{x_2y_1z_0}$ .



**Figure 5.** Rule 51254550<sub>8</sub>, 200 rows.



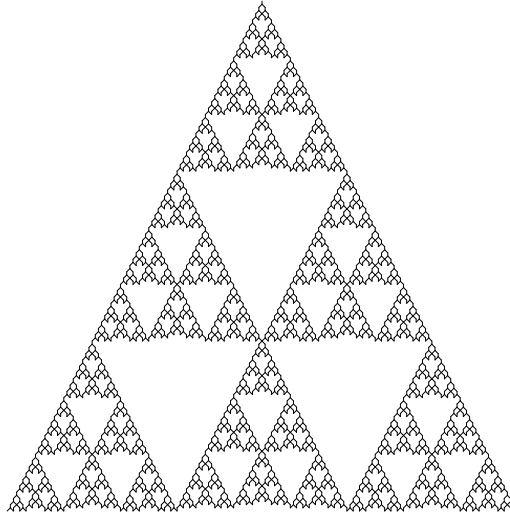
**Figure 6.** Complex pattern of GLCA, Rule 71055670<sub>8</sub>, 200 rows.

## 2 Symmetric fractal patterns

We can find all the pattern groups of Wolfram's classification (Class I-IV) among GLCA patterns otherwise the evolution of the patterns leads to homogenous, regular, chaotic and complex patterns. See *Figure 5* and *6*.

We show how can we realize *Pascal triangle modulo 3* symmetric fractal pattern by the monochromatic GLCA. See *Figure 7*.

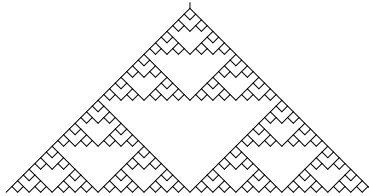
The *Sierpiński triangle* (Pascal triangle modulo 2) pattern can be realized in many ways. For example by applying an XOR binary operation or an iterated



**Figure 7.** Rule 00520520<sub>8</sub> of GLCA = Pascal triangle modulo 3.  
162 rows of a nested pattern = 4th approximation ( $2 \cdot 3^n$  rows of arcs).

function system (IFS) rule onto a binary square matrix. We get the same result with Wolfram's elementary cellular automaton [WE]. His simple rules 60, 102, 90 and 126 also give this pattern on different ways.

GLCA also gives other possibilities to realize this fractal pattern. The simplest one, Rule 00050550<sub>8</sub> can be drawn from any single root arc. The rule means draw two vertical branch arcs from single arcs and do not draw in other cases. See *Figure 8*. Rule 00020520<sub>8</sub>, 06523520<sub>8</sub> and 00720520<sub>8</sub> also make this fractal in another way.



**Figure 8.** Rule 00050550<sub>8</sub> of GLCA = Pascal triangle modulo 2.  
33 rows of a nested pattern = 5th approximation ( $1 + 2^n$  rows of arcs).

We can realize *Pascal triangle modulo 3* pattern in many different ways also. As an IFS fractal [W02], by recursive curves otherwise by Hamiltonian paths or Hamiltonian cycles [K17a,K17b]. With Wolfram's automaton we have to use more colours [W84,W02] (3 colours, totalistic rule, code 420) unlike my monochromatic GLCA pattern on *Figure 7*.

### 3 Searching for reversible rules

A reversible cellular automaton is a system that is deterministic in both directions in terms of time. It is also called invertible cellular automaton. In GLCA it means if we change the direction of all arcs of the mini trees into reverse we can continue the drawing at the other side of a root pattern. Most of the cases these directions belongs to different rule numbers. For reversible rules we have to find bijective pairs with the same rule number.

In reversible rules we have to avoid growing branches from nothing therefore the last digit of the rule is always equal to zero. We have to use assignments amongst root patterns and branch patterns with one-to-one correspondence. We have 7 different patterns so the maximum number of these unambiguous assignments are equal to the number of the permutations of our patterns:  $7! = 5040$ .

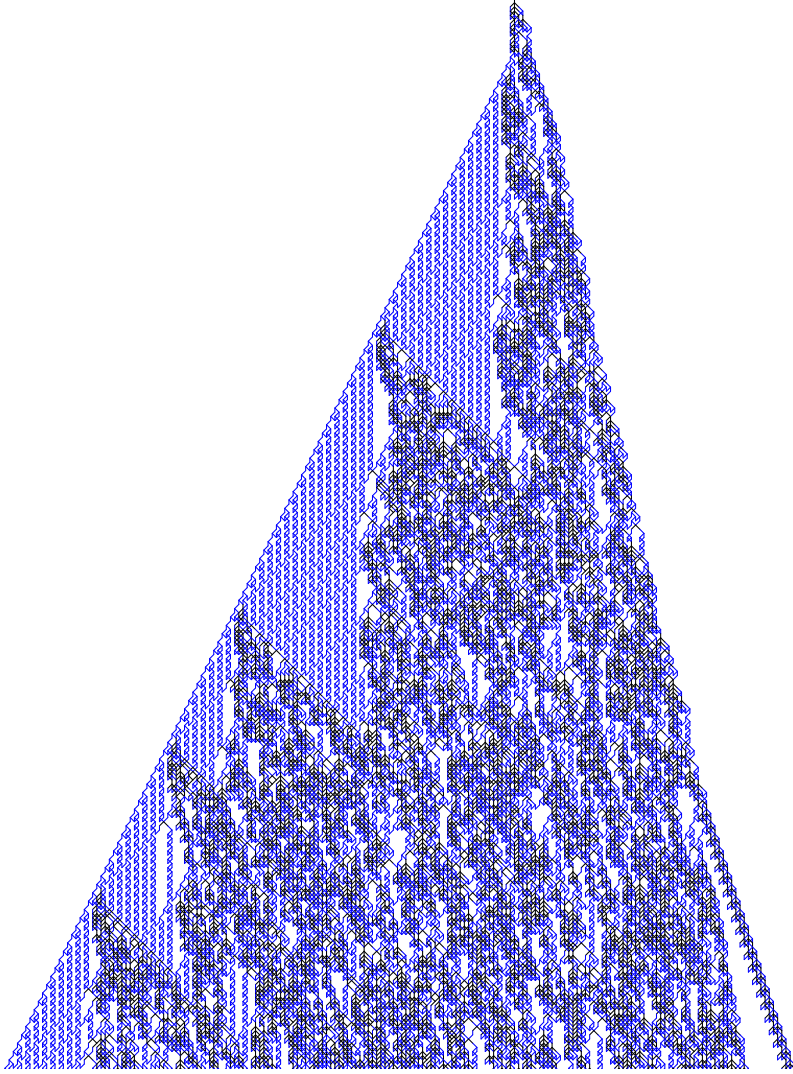
By leaving odd numbers of digits at their place-values in the octal rule number and changing the remaining digits pairwise by mutuality of the number of the place value and the correlating digit we get the following sum of binomials:  $\binom{7}{0} + \binom{7}{2} + 3\binom{7}{4} + 15\binom{7}{6}$ . In this case the rule number does not depend on the direction of the mini trees (assignments) therefore we get 232 different reversible rules. This is the number of the self-inverse permutations on 7 letters, also known as involutions [OEIS]. For example Rule 67234510<sub>8</sub> is a reversible one.

In the remaining cases we get a different rule number by changing the direction of the drawing. We get the correlating rule pair by replacing the digit values with the place-values of the rule number for example: Rule 35724160<sub>8</sub> and Rule 51637420<sub>8</sub> are correlating pairs.

### 4 Extensions and variations of the basic idea

By using the same 3 arcs long root and branch patterns and bichromatic arcs (2 drawing colours and 1 background colour) the triplets can be described as 3-digit long numbers in ternary numeral system. In this case we combine  $3^3$  root patterns with also 27 branch patterns and we have  $27^{27} = 3^{81}$  different rules. We can represent these numbers with 0 to 9 and A to Q symbols (as digits of numeral system 27). In this case the rule is a 27-digit long number consisting of these symbols. See *Figure 9*.

We recommend Wolfram's method the *totalistic rules* to define the assignments in an easier way. Instead of defining branch patterns for every possible root pattern it is enough to assign branch patterns to groups of root patterns as hues or densities of arcs. These hues or densities are equal to the sum of the digits of a root pattern. For example in monochromatic GLCA (1 drawing and 1 background colour) we have binary triplets as root patterns. The sum of the digits is between 0 and 3 therefore it is enough to define 4 assignments instead of 8. By using bichromatic arcs (2 drawing and 1 background colour) we have ternary triplets as root patterns. The sum of the digits is between 0 and 6 therefore it is enough to define 7 assignments instead of 27.



**Figure 9.** Rule  $HPD8962896DGH067K4MHQL013C0_{27}$   
Complex pattern in bichromatic version of GLCA.  
300 rows, root pattern is a single black vertical arc.



We can imagine GLCA on a fixed width space or on a cylindrical grid also.

We can colour and draw only the junctions instead of the arcs. It is a number triangle (from a single root pattern) or a number trapezoid (from a single row of root patterns). We have to use 8 colours to show the root patterns (junctions as data containers contain these octal values). Rule  $51254550_8$  on *Figure 1* and *Figure 5* makes the following *number triangle* of octal digits: 2, 104, 10504, 1042104, 105154504, 10430706104, ... etc.

The number triangle constructed as follows: for example root patterns 7, 4, 2, 1 are assigned with branch pattern 5. As a reverse binary number it is equal to  $1 + 0 + 4$ . From root pattern 2 we get 104, then from root pattern 104 we get:

$$\begin{array}{r}
 \underline{1\ 0\ 4} \\
 1\ 0\ 4\ .\ . \\
 .\ 0\ 0\ 0\ . \\
 \underline{.\ .\ 1\ 0\ 4} \\
 1\ 0\ 5\ 0\ 4
 \end{array}$$

We can make the 2D version of GLCA represented by arcs in 3D cubic space. Consider y axis as the growing direction otherwise the time. We get 9 possible arcs (1 vertical, 4 diagonal and 4 space diagonal arcs) at every grid point. By using monochromatic arcs we get  $2^9$  possible root patterns and the same 512 different branch patterns. It's worth to use a grouping of arcs to define the rules in an easy way. By using totalistic rules we have to summarize the number of arcs in an elementary pattern. In this case a pattern consists of 0 to 9 arcs therefore a totalistic rule contains 10 numbers between 0 and 511. These numbers symbolize a branch pattern for each density or hue of arcs. Beyond the natural monochromatic representation we can visualize every layer as a 2D animation as patterns change in time steps like a stroboscope. In this case it's worth to represent the coloured junctions as a square tessellation instead of the arcs. It could be a closer relative of *Conway's Game of Life*.

## 5 Summary

We have introduced our GraftalLace Cellular Automaton which makes a one-dimensional infinite monochromatic digraph otherwise an octal number triangle or number trapezoid by partly influences the states of the neighbour cells with bit operations. We have shown new ways to make known symmetric fractal patterns and unknown complex patterns. The monochromatic GLCA has  $8^8$  possible rules. We have chosen  $8^7$  rules in which none of the branches grow from nothing. We have found  $7!$  unambiguous rules in which 232 are reversible. We have shown possibilities to represent and extend our automaton in different ways. The 2D version of GLCA can be represented by a 3D graph in cubic space or as a 2D animated tessellation formed by the coloured junctions changing in time steps. It could be a closer relative of *Conway's Game of Life*. Beyond cryptographic utilization, the physical, chemical and biological connections might also be interesting.

## References.

- [W84] Wolfram, S.: *Computer Software in Science and Mathematics*, Scientific American, Vol. 251, Issue 3, September 1984.
- [W02] Wolfram, S.: *A New Kind of Science*, Wolfram Media Inc., 2002
- [D86] Dewdney, A. K.: *Computer Recreations — of fractal mountains, graftal plants and other computer graphics at Pixar*, Scientific American, Dec. 1986
- [PL90] Prusinkiewicz, P. and Lindenmayer, A.: *The algorithmic beauty of plants*, Springer, 1990.
- [WE] *Wolfram MathWorld / Elementary Cellular Automaton*,  
<http://mathworld.wolfram.com/ElementaryCellularAutomaton.html>
- [K17a] Kaszanyitzky, A.: *The generalized Sierpiński Arrowhead Curve*, 2017  
<https://arxiv.org/abs/1710.08480>
- [K17b] Kaszanyitzky, A.: *Triangular fractal approximating graphs and their covering paths and cycles*, 2017  
<https://arxiv.org/abs/1710.09475>
- [OEIS] Sloane, N.J.A.: *The On-line Encyclopedia of Integer Sequences*,  
<http://oeis.org/A000085>