Representation of fractal curves by means of L systems

Authors:

- Manuel Alfonseca

  Professor at the Universidad Autonoma of Madrid

  Address:

```
Escuela de Ingenieria Informatica
Universidad Autonoma de Madrid
Ciudad Universitaria de Cantoblanco
28049 Madrid SPAIN
```

  Telephone: (34) 1 397 4467

  FAX: (34) 1 397 xxxx

  Email: Manuel.Alfonseca@ii.uam.es
- Alfonso Ortega

  Ph.D. student at the Universidad Autonoma of Madrid

Abstract

Fractals can be represented by means of L-systems (Development Grammars), together with a graphic interpretation. Two families of graphic interpretations have been used: turtle graphics and vector graphics. This paper describes an APL2/PC system able to draw fractals represented by L-systems, with both graphic interpretations. A theorem has been proved on the equivalence conditions for both interpretations. Another point shown is the fact that supposed deficiencies in L-systems that have prompted proposals of extensions are really deficiencies in the graphic translation scheme.

Fractals

In 1890, the italian mathematician Giuseppe Peano described a certain curve, now known by his name, which displayed several **monstruous** properties, suchs as filling a surface (in the sense of passing through every point in it), and having a tangent at no point. Another monstruous line (the snowflake curve) was described in 1904 by Helge von Koch.

In 1919, H. Hausdorff introduced a new concept of dimension, different from the ordinary geometric dimension. According to his definition, later refined by Besicovitch, ordinary lines have a dimension of 1 and ordinary surfaces a dimension of 2, but the monstruous curves have a fractional dimension greater than 1. In fact, the Peano curve has a Hausdorff-Besicovich dimension of 2, while its value for the Koch snowflake is 1.2618595071429...

In 1975, Benoit B. Mandelbrot [Man 82] defined a fractal as an "object whose Hausdorff-Besicovitch dimension is greater than its geometric dimension". Fractals have special properties, such as self-similarity (containing copies of themselves) and underivability at every point. They are appropriate for the description of natural shapes.

There are three main kinds of fractals:

- Those obtained as the limit between the domains of convergence and divergence of a family of recursive mathematical equations, such as the *Mandelbrot set*.
- Those obtained by means of a transformation (the generator) applied recursively to an initial shape (the iniciator). This group includes the Peano and the Koch snowflake curves. Depending on the definition of the generator, it is possible to obtain deterministic, random or caotic fractals.

- Those obtained through the emulation of a Brownian movement.

In this paper, we are mainly interested in the second family of fractals.

L systems

In 1968, Aristid Lindenmayer [Lin 68] defined a new type of grammar (the parallel derivation grammar), which differs from the normal Chomsky grammars (sequential derivation grammars) because the grammar rules are applied simultaneously, rather than one at a time.

Parallel derivation grammars, also called L systems, can be classified in different ways:

- Context sensitive (IL systems) versus context free (0L systems).
- Deterministic (DL systems) versus non-deterministic.
- Propagative (PL systems) versus non-propagative.
- EL systems, with extensions.
- TL systems, with tables, where the set of production rules includes two or more complete subsets of 0L rules, that will be applied alternatively in each successive derivation.

These types may be combined: A D0L system is a deterministic context free system; a PD0L system is propagative, deterministic and context free; an EIL system is context sensitive with extensions; and so forth. A D0L system, for instance, is the three-fold (Σ, P, w), where Σ is an alphabet (a finite non-empty set of symbols); P is a set of production rules of the form A::=x, where $A \in \Sigma$ is a symbol in the alphabet and $x \in \Sigma^*$ is a (possibly empty) word or string of symbols in the alphabet; and $w \in \Sigma^*$ is the starting word or axiom. Every symbol appears exactly once at the left of a production rule (this makes the system deterministic).

A D0L scheme is the two-fold (Σ, P), of an alphabet and a set of production rules, and represents the family of all the D0L systems sharing those two components and differing in the axiom.

An example of a D0L system is:

```
    ( F,+,-, P, F++F++F )
```

where P is the following set of production rules:

```
        F ::= F-F++F-
        + ::= +
        - ::= -
```

A derivation of a word in a D0L system is the new word obtained when each symbol in the first word is replaced by the right part of the production rule whose left part is that symbol. In the previous example, we can get the following derivation from the axiom:

```
    F++F++F  →  F-F++F-++F-F++F-++F-F++F-
```

The word obtained becomes the starting point of a new derivation, and so on.

L systems have been successfully applied in the simulation of biologic processes, such as plant growth, leaf development, pigmentation of snail shells, etc. They are also very appropriate to represent fractal objects obtained by means of recursive transformations [Dem 85, Pru 86, Gie 91, Cul 91a, Cul 91b]. The initiator maps to the axiom of the L system, the generator to the production rules, and the recursive applications of the generator to the initiator correspond to the successive derivations of the axiom.

Something else is needed, however: a graphic interpretation that makes it possible to convert each word generated by the L system into a visible curve. The fractal object is the limit of the sequence of curves generated by the L system with the appropriate graphic interpretation.

It is very important to separate the L system from the graphic interpretation. Otherwise, a deficiency in the latter may be mistaken for one in the former. In the past, different extensions have been proposed for L systems, supposedly incapable of generating certain fractal objects, which a different graphic interpretation would make it possible to obtain.

Two different families of graphic interpretations of L systems have been used:

- Turtle graphics
- Vector graphics

Turtle graphics

Created by Seymour Papert in 1980, the graphic is the trail left by a moving invisible "turtle", with a state defined by its position and the direction in which it is pointing. The state of the turtle may change as it moves a step forward, or as it rotates a given angle in the same position.

In the simplest turtle graphics interpretation, the alphabet of a D0L system consists of three symbols:

```
Σ = { F, +, - }
```

The graphics interpretation of a word is as follows:

F	The turtle moves one step forward, in the direction it is pointing, leaving a visible trail. We will call F a *drawing symbol*.
+	The turtle rotates a positive angle α
-	The turtle rotates a negative angle α

Additional rules complicate the turtle graphics and make it possible to generate more complicated fractals. Between those rules, we will mention the following:

- Upper case letters different from F have no graphic representation and leave the state of the turtle unchanged. In the following, we will call such letters *non-graphic symbols*.

- Lower case f makes the turtle move a step forward, with no visible trail. In the following, we will call letters such as f *moving symbols*. This extension makes it possible to represent unconnected fractals.

- An open parenthesis pushes the state of the turtle into a stack; a closed parenthesis pops the top of the stack and restores the turtle state. This extension makes it possible to represent branching fractals.

- Additional letters may be defined to act the same as F and f (*drawing symbols* and *moving symbols*).

- Symbol ! makes the turtle rotate 180 degrees.

- { Braces } indicate that the area enclosed in the braces must be filled.

- Symbol % followed by a number changes the color of the trail left by the turtle.

A given fractal may be represented by means of three components: an L system, a turtle graphic interpretation (made up of a combination of the preceding rules), and an angle step.

Vector graphics

In this family of interpretations, every symbol in the alphabet of the L system is associated to a vector in a rectangular Cartesian system. A word (a string of symbols) is represented by the catenation of the vectors of the symbols that make the word.

In the simplest case, a fractal may be defined by two components: an L system, and a mathematical application V: $\Sigma \rightarrow R^2$ (the vector interpretation). We assume that all the vectors associated to the symbols in the alphabet generate visible movements. The graphic representation of each derivation of the L system is a set of straight connected segments.

This vector interpretation is capable of representing branching fractals, if for every symbol in the alphabet there is another associated to the opposite vector, which makes it possible to return to the start of the branch. However, a different vector interpretation is needed to build unconnected fractals. These can be covered by means of the following mathematical application: V': $\Sigma \rightarrow 0,1 \times R^2$. In this case, the vector equivalent to a symbol has three elements instead of two: a visibility coefficient (0 or 1) is added, indicating that the vector displacement should be visible (1) or invisible (0).

An APL2 implementation

A system has been implemented in APL2/PC to experiment with fractals using different graphic interpretations. A fractal scheme is represented by a 3-element vector (additional elements in vectors are considered to be comments):

[1] A [T][P][D]0L scheme.

[2] A graphic representation.

[3] Additional information, depending on the graphic representation.

The 0L scheme is represented by the set of production rules (a matrix of two columns), from which the alphabet may be deduced (it is the set of all the symbols appearing in the rules). The rules have the following structure:

- For a D0L system: all the rules are of the form

```
A (x)
```

where A is a symbol in the alphabet, x is a word and the parenthesis represent general array depth. The set of all the rules is a two-column matrix.

- For a PD0L system: none of the rules is of the form

```
A ()
```

i.e., its right part is the empty word.

- For a non-deterministic 0L system: at least one of the rules is of the form

```
A ((x y ...))
```

where x, y, ... are words.

- For a DT0L system: all the rules are of the form

```
A (x1 x2 ...)
```

where x1, x2, ... are words.

- For a non-deterministic T0L system: all the rules are of the form

```
A ((x1 y1 ...) (x2 y2 ...) ...)
```

where x1, y1, ... are words.

At this time, the graphic representation may be one of the following:

- Vector graphics.
- Turtle graphics with the five first extensions.
- Pixel graphics (a special case of vector graphics where only the extreme point of each vector is visible).
- Turtle graphics with all the extensions mentioned above.

The additional information is the following:

- For vector and pixel graphics: a matrix containing the definition of the vectors for all the symbols at the left of the rules in the 0L scheme.
- For turtle graphics: the angle step and the sets of drawing symbols and moving symbols (the set of non-graphic symbols may be deduced from those). If the sets are not given, they are assumed to be F and f, respectively.

The system includes a set of functions performing the following actions:

- Interactive creation of a fractal scheme (function MAKE0L). Rules are written with the following syntax:

  ```
  A x1|y1|...[ x2|y2|...]
  ```

  to represent all kinds of [T][P][D]0L schemes.
- Derivation of a word by a given 0L scheme (function D0L).
- Successive derivations of a word by a 0L scheme (function TEST0L).
- Drawing a given step of a fractal curve from a 0L scheme and an axiom (function VER).
- Drawing successive steps to a fractal curve from a 0L scheme and an axiom (function TESTDRAW).

The fractal schemes created may be kept in a file of APL objects using the AP211 auxiliary processor. The steps to the fractal curve are drawn using the AP207 VGA graphics auxiliary processor.

Examples

The examples given below have been designed by different authors and are common in the literature.

- The PD0L scheme

  ```
  F ::= F-F++F-F
  + ::= +
  - ::= -
  ```

  with axiom F++F++F, a turtle graphic interpretation, and a step angle of 60 degrees, generates the Koch snowflake curve, whose fifth iteration appears in figure 1.

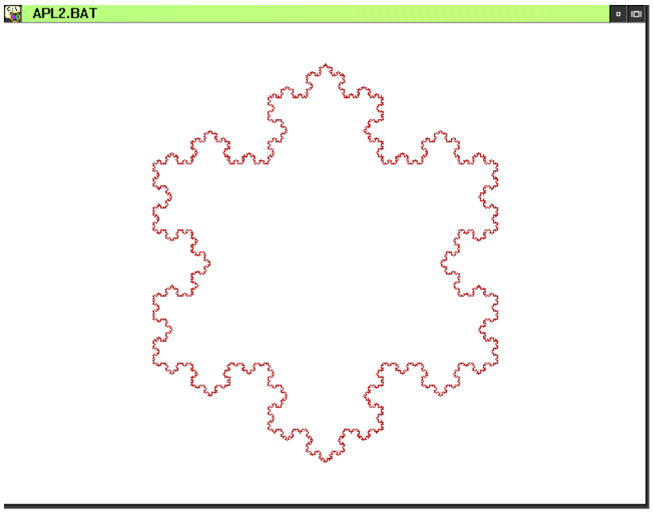

Figure 1

In our system, it is not necessary to give the rules for symbols in the set +, -, (,), !, , , which usually remain invariable. This means that the last two rules in the preceding L-scheme may be omitted. In the following examples, these rules will not be written, but are always assumed to be there, as they will be automatically added by the program.

- The preceding fractal may also be generated by the following PD0L scheme

```
A ::= AFBA
B ::= BACB
C ::= CBDC
D ::= DCED
E ::= EDFE
F ::= FEAF
```

with axiom ACE, a vector graphic interpretation and the following two column vector definition:

Symbo l	X	Y
A	1	0
B	0.5	r
C	-0.5	r
D	-1	0
E	-0.5	-r
F	0.5	-r

where r is one half of the square root of 3.

- The PD0L scheme

```
A ::= AF+F+BF-F-AF-F-BF-F-AF+F+BF+F+AF+F+BF-F-A
B ::= BF-F-AF+F+BF+F+AF+F+BF-F-AF-F-BF-F-AF+F+B
F ::= F
```

where A, B is the set of non-graphic symbols, with axiom AF, a turtle graphic interpretation, and a step angle of 45 degrees, generates the Peano plane-filling curve, whose third derivation appears in figure 2.

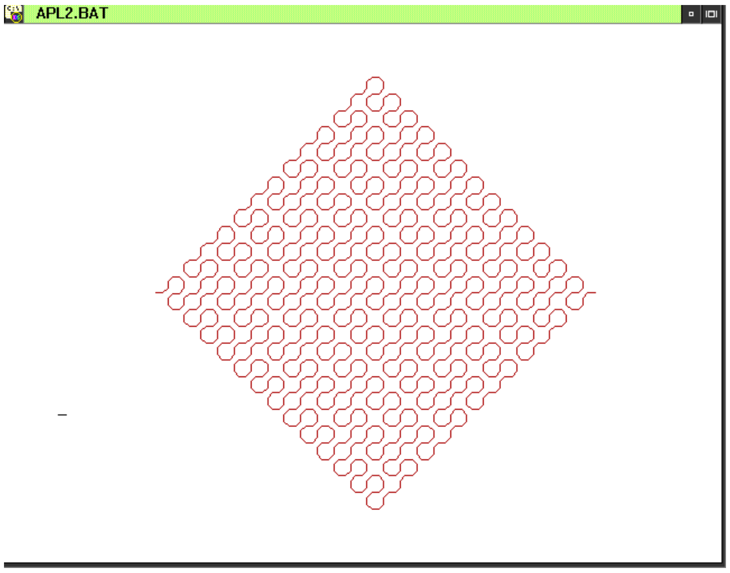

Figure 2

- The PD0L scheme

```
A ::= FQAPASH            P ::= P
B ::= GRBQBPE            Q ::= Q
C ::= HSCRCQF            R ::= R
D ::= EPDSDRG            S ::= S
E ::= DSEPEQB
F ::= APFQFRC
G ::= BQGRGSD
H ::= CRHSHPA
```

with axiom A, a vector graphic interpretation and the following two column vector definition:

Symbol	X	Y	Symbol	X	Y
A	0	0	G	0	0
B	0	0	H	0	0
C	0	0	P	1	0
D	0	0	Q	0	1
E	0	0	R	-1	0
F	0	0	S	0	-1

generates the Hilbert curve whose sixth derivation appears in figure 3.

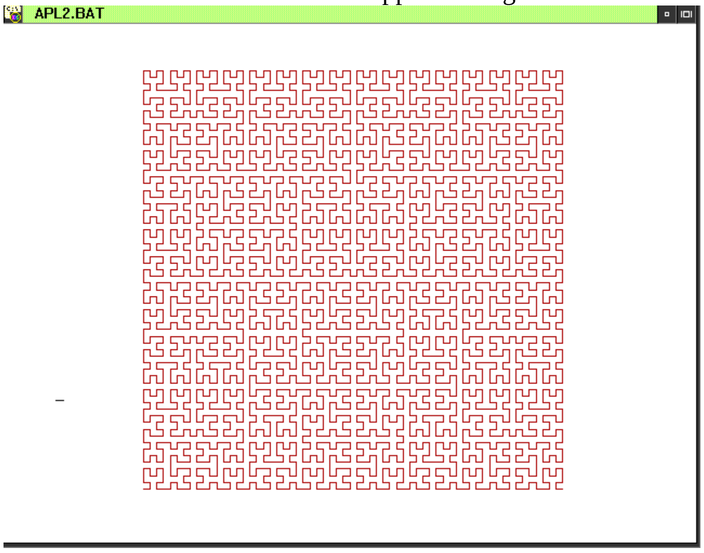

Figure 3

- The PD0L scheme

```
F ::= FfF
f ::= fff
```

with the default sets of drawing and moving symbols, axiom F, a turtle graphic interpretation, and a step angle of any value, generates the Cantor set of the third order, whose axiom and first five derivations appear in figure 4.

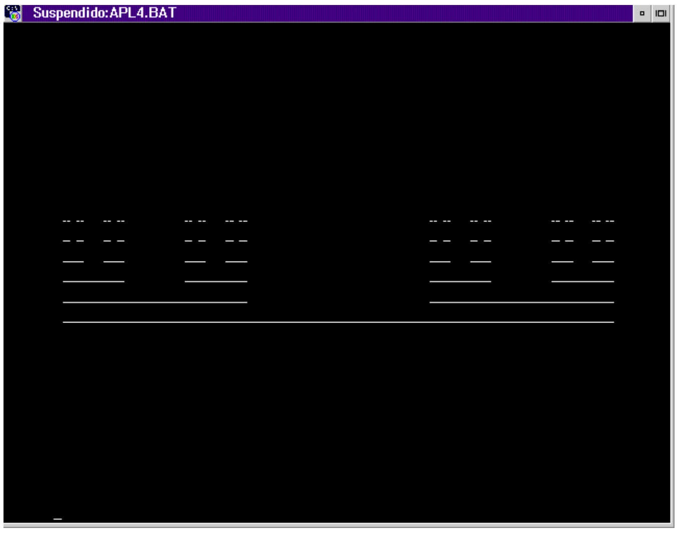

Figure 4

- The preceding fractal may also be generated by the following PD0L scheme

```
A ::= ABA
B ::= BBB
```

with axiom A, a vector graphic interpretation and the following three column vector definition:

Letra	**Visibility**	**X**	**Y**
A	1	1	0
B	0	1	0

- The PD0L scheme

```
F ::= FF+(+F-F-F)-(-F+F+F)
```

with axiom F, a turtle graphic interpretation, and a step angle of 22.5 degrees, generates the branching fractal curve, whose fourth derivation appears in figure 5.

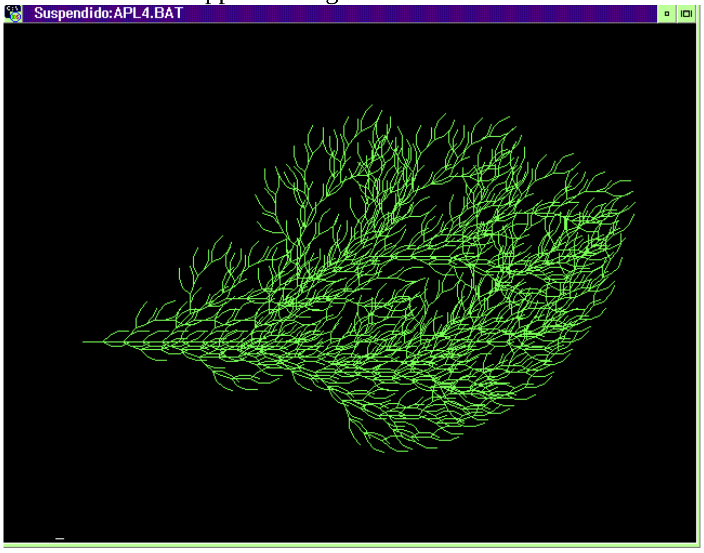

Figure 5

- The PD0L scheme

```
A ::= F-(A+A)+F(+FA)-A
F ::= FF
```

where A, F is the set of drawing symbols, with axiom A, a turtle graphic interpretation, and a step angle of 22.5 degrees, generates the fractal whose fifth derivation appears in figure 6. This is very similar to the fractal that moved P. Prusinkiewicz [Pru 86] to propose an extension to 0L systems (pL systems), assuming that the former were not powerful enough to handle it. In fact, Prusinkiewicz fractal can be obtained by enclosing every A between additional parentheses in the preceding PD0L scheme. This indicates that his assumption was unwarranted, as an appropriate extension to the graphic interpretation is sufficient to overcome the difficulty.

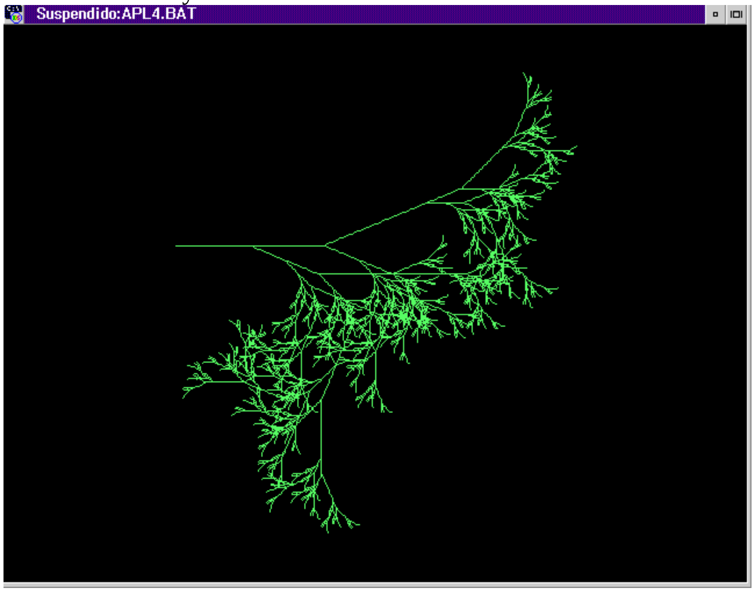

Figure 6

- The PD0L scheme

```
F ::= FF
G ::= G
L ::= (+G-G-G+!+G-G-G)
R ::= F(--L)(++L)F
T ::= R+(T)--(--L)R(++L)-(T)++T
```

where F, G is the set of drawing symbols, L, R, T is the set of non-graphic symbols, with axiom T, a fully extended turtle graphic interpretation, and a step angle of 30 degrees, generates the fractal whose fourth derivation appears in figure 7. This is another of the fractal curves supposedly requiring an extension to D0L systems.

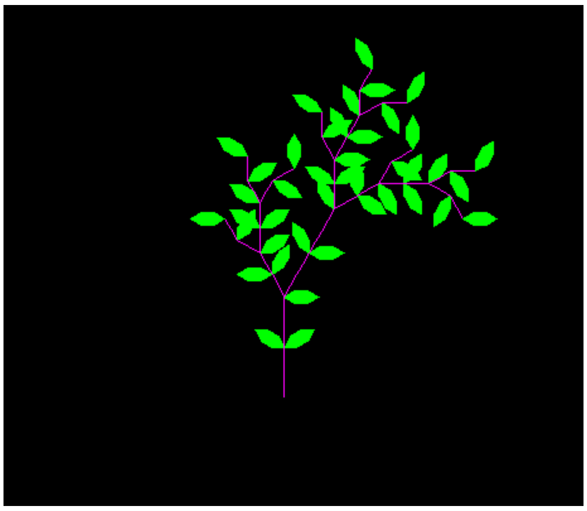

Figure 7

- The P0L scheme

```
F ::= F-F++F-F | F+F--F+F
```

with axiom F++F++F, a turtle graphic interpretation, and a step angle of 60 degrees, is a non-deterministic variant of the Koch snowflake curve. A possible fifth iteration (depending on the random seed used) appears in figure 8. If the first alternative were always applied, we would obtain the snowflake curve of figure 1. The second alternative would generate a variant of the same curve, with the direction of the generator inverted. It may be seen that this drawing is reminiscent of the coastline of an island or a continent. However, it is a little too complicated, due to the fact that its Hausdorf-Besicovich dimension is (approximately) 1.26, while the dimension of real coastlines uses to belong to the interval 1.15 to 1.25.

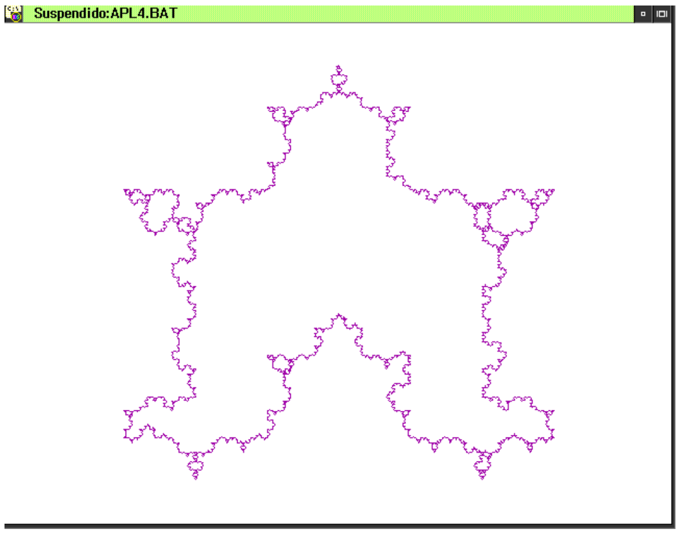

Figure 8

- The TP0L scheme made by the two sets of rules:

```
A ::= AFBA        A ::= ABFA
B ::= BACB        B ::= BCAB
C ::= CBDC        C ::= CDBC
D ::= DCED        D ::= DECD
E ::= EDFE        E ::= EFDE
F ::= FEAF        F ::= FAEF
```

where the first table is the set of productions used above to generate the Koch snowflake, while the second table inverts the direction of the fractal generator, produces a curious variant of the snowflake whose fifth iteration may be seen in figure 9.

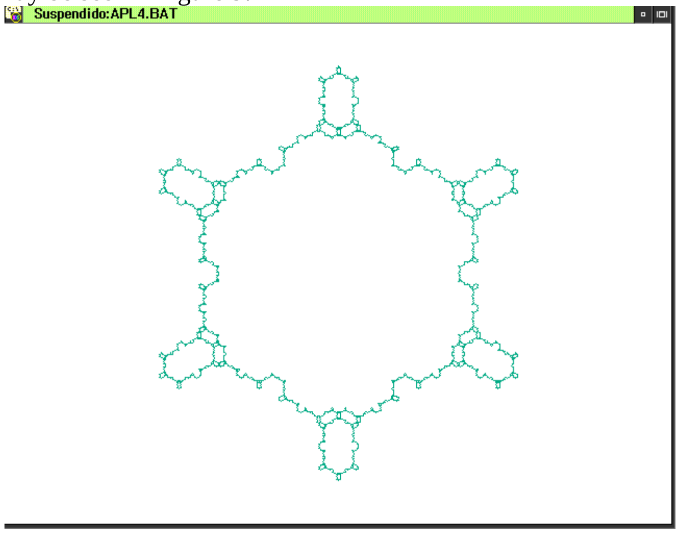

Figure 9

- Finally, the PD0L scheme

```
A ::= ABB
B ::= BC
C ::= CD
D ::= DAC
```

with axiom CCCC, a vector graphic interpretation and the following two column vector definition:

Letra	X	Y
A	8	0
B	0	0.75
C	-4	0
D	0	-0.75

generates a fractal curve whose tenth derivation is uncannily similar to a human hand (see figure 10).

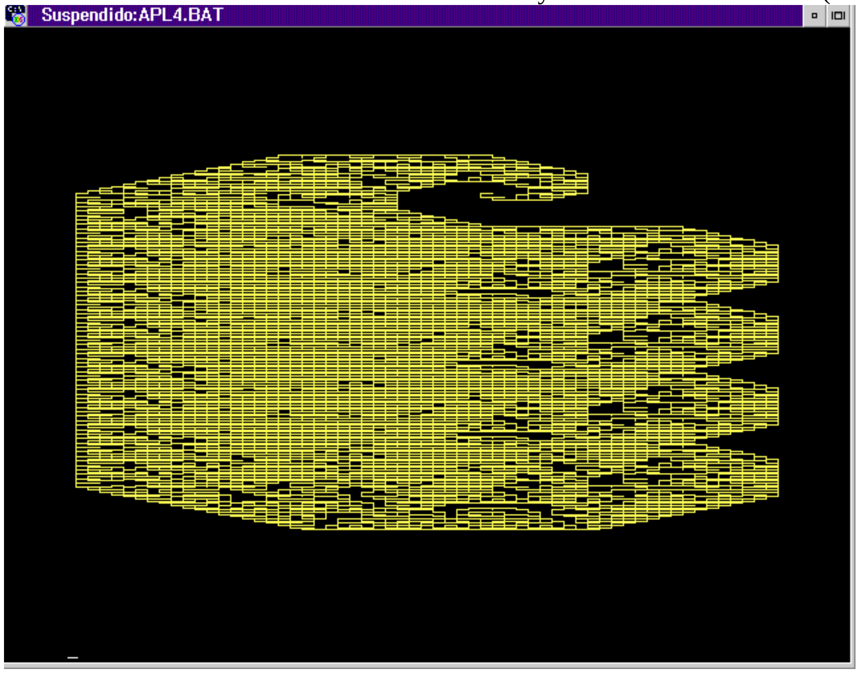

Figure 10

An equivalence theorem

Since we have two different graphic interpretations for L systems (turtle graphics and vectors) the question of the possible equivalence of both interpretations springs to mind. In other words: given a fractal represented by an L system with turtle graphics, can we represent the same fractal by another (usually different) L system with vector graphics? In fact, in the preceding examples, we have shown that the same fractal may be obtained by means of totally different L systems, each of which uses a different graphic interpretation.

We have proved a theorem that can be considered as a first approach to the demonstration of full equivalence, which states:

1. *For every fractal curve represented by a D0L system with a vector graphics interpretation, there is another D0L system with a turtle graphics interpretation that represents the same fractal curve.*

2. *For every fractal curve represented by a D0L system with a turtle graphics interpretation, there is another D0L system with a vector graphics interpretation that represents the same fractal curve, if the first representation satisfies the following restrictions:*

    - *There are two integers, n and k, such that n.&alpha. = 2.k.&pi., where &alpha. is the turtle incremental angle.*
    - *For every rule in the L system, the number of + symbols minus the number of - symbols is a multiple of n, where n is the constant defined in the preceding condition.*
    - *The alphabet for the L system includes the + and - signs, together with three sets of letters, corresponding to draw, move, and non-graphic symbols.*

In a computer, the first condition is always in effect, since the turtle incremental angle is a rational number (one cannot store irrationals in a computer). Therefore, in practice, the only restrictions are the last two. The theorem, such as stated, seems to imply that turtle graphic interpretations are more powerful than vector graphics.

Our proof of the theorem is a constructive algorithm that, starting from an L system of one kind, generates an equivalent system of the other kind. In fact, the vector graphics L-system that generates the Koch snowflake curve shown in the examples, was obtained from the equivalent turtle graphics L-system, using our algorithm.

Conclusion

We have seen that D0L systems, given an appropriate graphic interpretation, are quite powerful to represent large families of fractal curves, even some that previous authors had assumed required nonstandard extensions.

It is thus important to isolate the D0L system from its graphic interpretation, which can be very different and belong either to the turtle or the vector family.

We have also proved a fractal equivalence theorem between L systems associated with a subset of turtle graphics and those associated with vector graphics.

References

[Cul 91a] K. Culik II, S. Dube, "Balancing Order and Chaos in Image Generation". *Proc. 18th Int. Coll. on Automata, Languages and Programming*, ed. J.L. Albert, B. Monien, M.R. Artalejo, Springer-Verlag, Berlin, 1991, pp. 600-614.
[Cul 91b] K. Culik II, S. Dube, "New Methods for Image Generation and Compression", *Proc. Conf. on Facts and New Trends in Computer Science*, ed. H. Maurer, Springer-Verlag, Berlin, 1991, pp. 69-90.
&lberk.Dem 85] S. Demko, L. Hodges, B. Naylor, "Construction of Fractal Objects with Iterated Function Systems". *Computer Graphics*, Vol. 19:3. pp. 271-278, 1985.
[Gie 91] E.G. Giessmann, "Generation of fractal curves by generalizations of Lindenmayer's L-systems". *Proc. 1st IFIP Conf. on Fractals in the Fundamental and Applied Sciences*, ed. H.O. Peitgen, J.M. Henriques, L.F. Penedo, North Holland, Amsterdam, 1991, pp.147-157.
[Lin 68] A. Lindenmayer, "Mathematical models for cellular interactions in development" (two parts). *J. Theor. Biol.* 18, pp. 280-315, 1968.
[Man 82] B.B. Mandelbrot, *The Fractal Geometry of Nature*. W.H.Freeman, San Francisco, 1982.
[Pru 86] P. Prusinkiewicz, "Graphical Applications of L-Systems", *Proc. Graphics Interface 86 and Vision Interface 86*, ed: M. Wein, E.M. Kidd, Vancouver, Canada, 1986, pp. 247-253.