serendipity

Not only computing – also art

JOHN LANSDOWN

Production rules – OK?

Over the past few issues we've been looking at examples of computer graphics from various universities and colleges. This time the works are from Grahame Andrew and some of his students for the Diploma in Art and design at the Sir John Cass School of Art in London. The drawings (Figures 1 to 4) are output from Grahame's BBC Micro program which uses production rules to generate its results.

Although production rules were first described as methods for logical manipulation by the American mathematician, Emil Post, as far back as 1943, it is only recently that they have become very well-known. Their present popularity arises because they feature as the main knowledge representation technique in many expert systems. In such systems, production rules usually take the form

IF (some antecedent) THEN (some consequent).

In this way they can be used to encapsulate knowledge about all sorts of things. For example, in the context of car maintenance:

IF exhaust is backfiring
AND dirty
THEN suspect that the mixture is too rich.

Alternatively, in the context of the Rita system for assisting users to make interfaces to external sources of data:

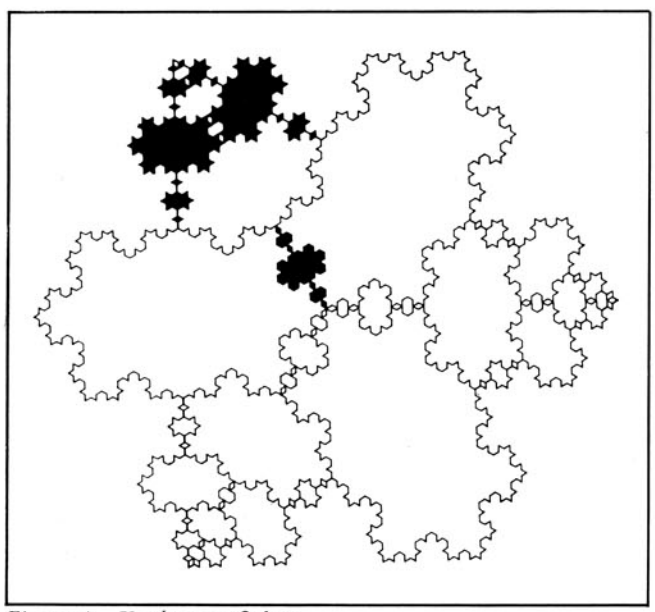


Figure 1 Koch snowflakes

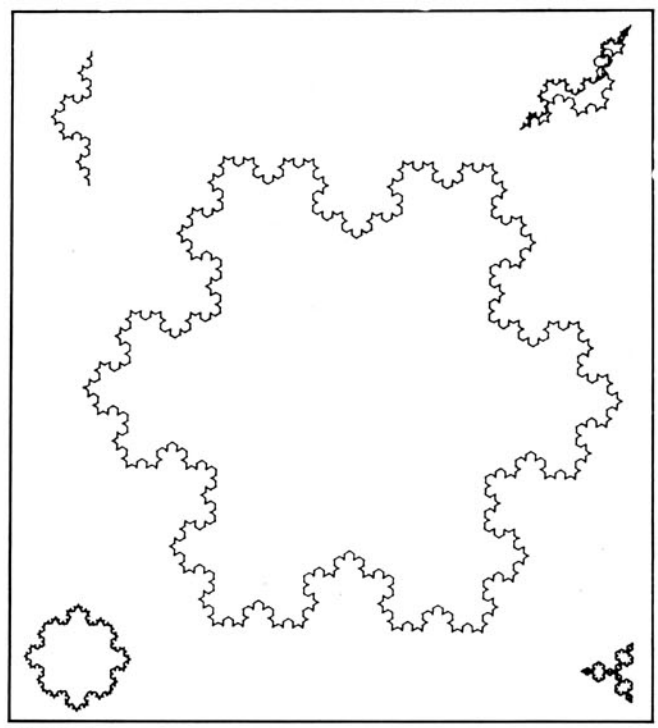


Figure 2

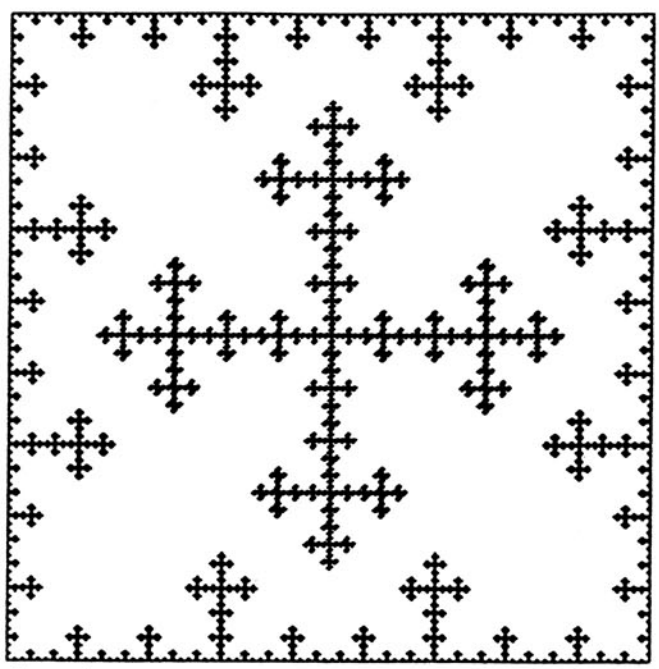


Figure 3

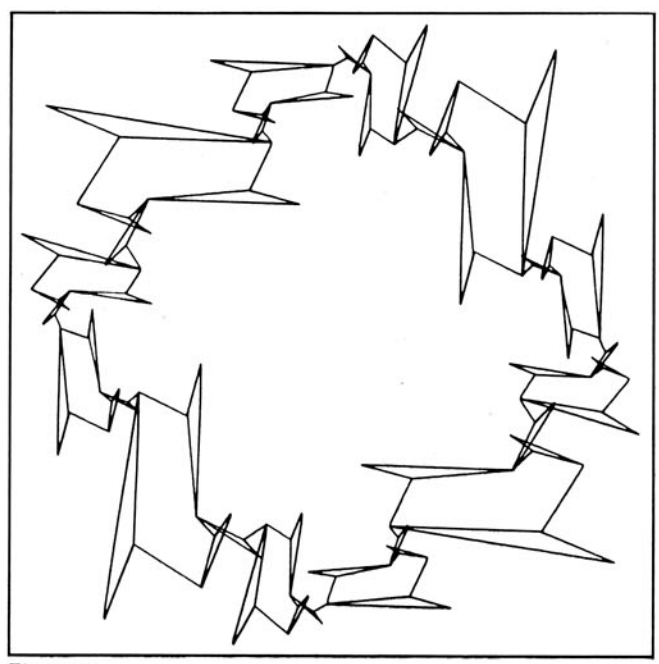


Figure 4

IF the type of person is 'Arpanet user'
AND location of person is 'Stanford AI Lab'
THEN set the net address of the person to 'SU-AI'

However, in their basic form, production rules can be seen simply as instructions for substituting one set of symbols for another. These symbols may be words or sentences in a natural language as in the above cases or more abstract concepts such as graphical or logical entities. In the latter situation, we can think of a production rule such as that in Figure 5 as saying, 'Wherever you see the element on the left-hand side of the arrow, substitute the element on the right-hand side'.

Figure 5 Snowflake production rule

Using this interpretation (which, in fact, is closer to Post's original formulation than the more well-known one), we can create designs of all sorts.

The usefulness of Grahame's program is that it lets users set up their productions quickly and easily and provides a ready mechanism for interpreting them. (In passing, it is worth mentioning Grahame's Master's thesis done last year at the Department of Design Research at the Royal College of Art. In this he outlines some of the techniques available for computer representation of multivariate data. It is well worth reading not least for the way in which it sets the graphics methods into historical perspective.)

A return to fractals

If we choose our productions according to certain criteria – details of which are too complex to go into here – we can produce fractal lines such as Koch, Sierpinski and Peano curves. Apart from all their other features, these curves have the curious property that, like all lines, their topological dimension is 1 but, unlike Euclidean lines, as they grow more complex in their weaving in and out without crossing themselves, they visit more and more points in the plane – which has a topological dimension of 2. This is one of the properties that led Benoit Mandelbrot to adopt his definition of 'fractional dimensionality' from which the word 'fractal' was coined.

Fractal lines have dimensionality between 1 and 2; fractal surfaces (such as mountains) have dimensionality between 2 and 3; fractal volumes (such as clouds) have dimensionality between 3 and 4. Furthermore, if we take a fractal surface of a certain dimensionality and slice it through horizontally, the dimensionality of the surrounding edge is 1 less than that of the whole. Thus an island whose surface has a fractal dimension of, say, 2.32 has a coastline of dimension 1.32.

Snowflakes and other pictures

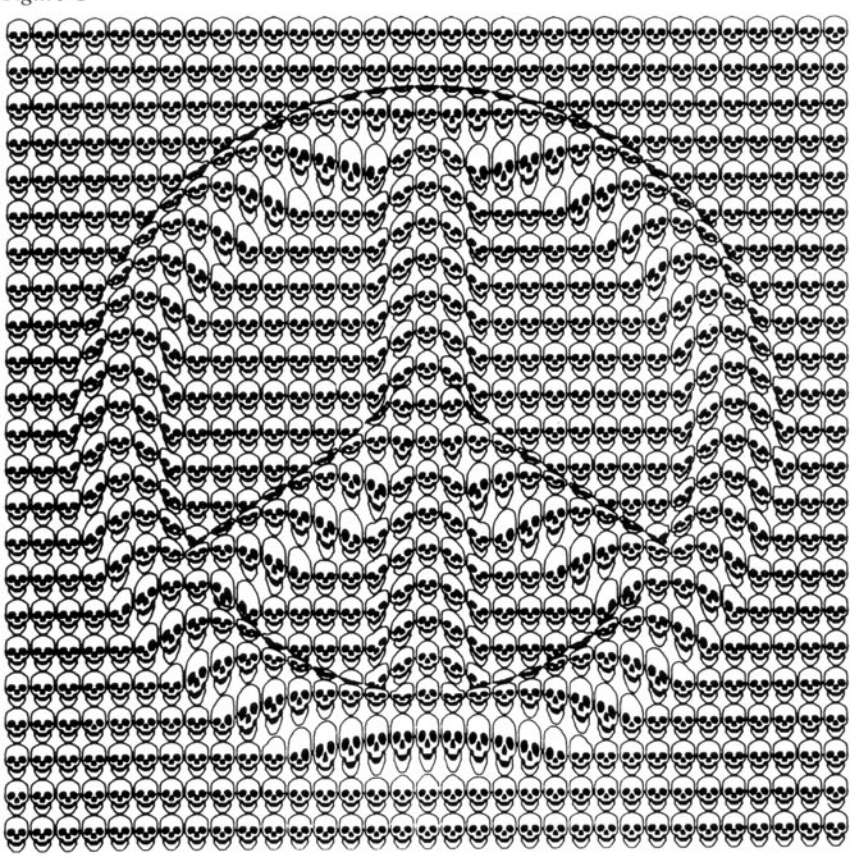
Of course, we can generate non-stochastic fractal curves like the Koch snowflake without resorting to production rule representations. A set of recursive programs for drawing such curves is given in Chapter 6 of that excellent book by Jim McGregor and Alan Watt called, *The Art of Microcomputer Graphics for the BBC Micro/Electron* (Addison-Wesley 1984). Even if you don't have a BBC Micro or Electron, you will find these programs, and the others given in the book, of great interest. Incidentally, they also deal with the Escher-like patterns we looked at in the March 1985 issue. Addison-Wesley also publish a BBC Micro disc called 'Tesselator' which contains some very nifty techniques for interactively creating such patterns.

T.B.

Computer art as propaganda

Artists have long used their skills to draw our attention to the social and political issues of the day – indeed it could be (and is often) argued that this is one of the primary functions of art. Computer artists have been as active as others in this area. Figure 6 shows a telling but, to my mind, slightly ambiguous image in this genre by Alex King who is now a Tutor and Researcher at The Dorset Institute of Higher Education although the drawing was done when he was at Bristol. There are some potentially important graphical developments going on at Dorset in conjunction with their newly approved Communication and Media course. I hope to include something about these in a future issue.

Figure 6



Dear Sir...

continued

There are problems, of course. I find it a tremendous waste of intellectual resources that there are no full time support staff for this committee, but this seems to stem from the rules of ANSI, under which they operate. And I agree that there is an overemphasis on backwards compatibility. However, despite my sympathy for the 'small is beautiful' design philosophy, I find that it is an interesting discipline for the proposer of modifications to Fortran to have to convince this mixture of developers and users that the idea is well founded.

Languages are born, grow up, give birth, mutate, grow old, and sometimes even die. If the users and manufacturers who are investing in the preparation of a new Fortran standard are misguided then this standard will flop. If Ada demonstrates adequate performance in an area that it has not been specifically designed for, then perhaps Fortran 8X compilers will become collector's items. Like quite a few languages designed by a small team rather than by a committee.

DAVID O. WILLIAMS
Geneva