Designing with L-Systems, Part 5: Termination

Most L-Systems produce longer and longer strings with each successive generation, and do this endlessly. This is intentional in the design of such L-Systems, where successive generations produce more complex and detailed patterns. Generation goes on endlessly because the rewriting rules contain variables [1]. Such L-Systems are called *nonterminating*.

It is possible to design nonterminating L-Systems that "loop" and have only a fixed number of different generations. A simple example is

seed: X
rules: X → Y
Y → X

where the generations are:

X
Y
X
Y
…

Such L-Systems are contrived aberrations and are not interesting for design purposes.

It is also possible to design L-Systems in which generation leads to a string with no defined variables. In this case, all subsequent generations would be the same, and generation effective terminates. Such L-Systems are called *terminating*.

An example of a terminating L-System is

seed: X
rules: X → YY
Y → ZaZ

where the generations are

X
YY
ZaZZaZ

In this case, different strings can be provided for Z during interpretation to give different results.

Although terminating L-Systems are limited in the variety of patterns they can produce, they are nonetheless useful in design.

Consider, for example, this L-System:

seed: X
rules: X → Y, 1, 2, 3, Y

Generation quickly terminates with the string

Y, 1, 2, 3, Y

If a rule for Y is added

Y → 4, 3, 2

the result is

4, 3, 2, 1, 2, 3, 4, 3, 2

On the other hand, if

Y → 1, 2, 3, 4, 3, 2

the result is

1, 2, 3, 4, 3, 2, 1, 2, 3, 1, 2, 3, 4, 3, 2

Put in words, this L-System characterizes all strings that have two instances of a given string separated by 1, 2, 3. This is, of course, obvious. But the idea can be used as a design tool.

For example, the next step might be to provide a rule for Y that contains a variable:

Y → 4, 3, 2, Y

This results in a nonterminating L-System with endless generation:

X
Y, 1, 2, 3, Y
4, 3, 2, Y, 1, 2, 3, 4, 3, 2, Y
4, 3, 2, 4, 3, 2, Y, 1, 2, 3, 4, 3, 2, 4, 3, 2, Y
4, 3, 2, 4, 3, 2, 4, 3, 2, Y, 1, 2, 3, 4, 3, 2, 4,
3, 2, 4, 3, 2, Y
…

April 10, 2004; last revised August 1, 2004

Reference

1. *Designing with L-Systems, Part 4: Articulated L-Systems*, 2004:
http://cs.arizona.edu/patterns/weaving/webdocs/gre_ls04.pdf

Ralph E. Griswold
Department of Computer Science
The University of Arizona
Tucson, Arizona

© 2004 Ralph E. Griswold