CPS 140 - Mathematical Foundations of CS
Dr. S. Rodger
Section: Modeling using L-Systems (handout)



**L-system**

L-systems are grammatical systems introduced by Lyndenmayer to describe biological developments such as the growth of plants and cellular organisms.

An L-system is composed of three parts $(\Sigma, h, w)$

| | | |
|---|---|---|
| $\Sigma$ | finite alphabet | set of symbols |
| h | rewriting rules | each symbol is replaced by string of symbols |
| w | axiom | starting point |

h is finite substitutions, h:$\Sigma \rightarrow \Sigma^*$.

h(w) is computed by replacing every symbol in w that has a rewrite rule by that rule.

A language L of an L-system is the word sequence generated by

- $h^0(w) = w$
- $h^1(w) = h(w)$
- $h^2(w) = h(h(w))$
- ...

L = $\{h^i(w) \mid i \geq 0\}$

NOTE: If h(a)=bb we will write this as a rule

a ==> bb

meaning the symbol a can be replaced by the symbols bb.

**Example:**

| | | |
|---|---|---|
| $\Sigma$ | alphabet: | $\{a, b\}$ |
| h | rules: | a ==> a |
| | | b ==> ab |
| w | axiom: | ab |

Notes:

- ==> means "is replaced by"
- left hand side of rule must be a single character
- there is at most one rule for each character

What is the language L of strings represented by this L-system? that is, starting with the axiom, what are all the strings that can be generated from the rules?

L =

**Example:**

| | | |
|---|---|---|
| $\Sigma$ | alphabet: | $\{a\}$ |
| h | rules: | a ==> aa |
| w | axiom: | a |

L =

**Drawing a picture of an L-system**

Defining an L-system: (3 parts in this order)

- Axiom definition: This must be the first line of the file
- Production rules: Defines the replacement rules.
  There must be spaces between the symbols on the right hand side of rules.
- Geometric rules: Defines colors, widths, etc.

Symbols for drawing and moving:

- g: draw a line one step in the current direction
- f: move forward one step in the current direction

**Example:**

This example is in the file: samp1

| | |
|---|---|
| SET axiom X | axiom definition |
| X ==> g f g | production rule |
| | geometric rules |
| SET d 15 | length of line drawn is 15 units |
| SET iniwidth 5 | width of initial line is 5 units |
| SET color black | |

**Example:**

This example is in the file: samp2



| | |
|---|---|
| SET axiom X | start symbol is X |
| X ==> g f g X | only change from previous program, repetition |
| SET d 15 | length of line |
| SET iniwidth 5 | width of initial line |
| SET color black | |

L =

**Symbols for changing direction**

- +: change direction to the right in a determined angle
- −: change direction to the left in a determined angle
- &: change direction pitch down in a determined angle
- ∧: change direction pitch up in a determined angle
- ∗: change direction roll left in a determined angle
- /: change direction roll right in a determined angle
- %: change direction 180 degrees

**Example:**

This example is in the file: samp3

| | |
|---|---|
| SET axiom X | start symbol is X |
| X ==> g g g X + Y<br>Y ==> g | rule with 2 replacements |
| SET d 15 | length of line |
| SET iniwidth 5 | width of initial line |
| SET angle 15 | angle for change of direction |
| SET color blue | initial color |

L =

**Example:**

We will make just a slight change in the L-system. This example is in the file: samp4

SET axiom X

X ==> g change blue g g change yellow + Y X — only difference with file samp3
Y ==> g

SET d 15 — length of line
SET iniwidth 5 — width of initial line
SET angle 15 — angle for change of direction
SET color black — initial color

L =

**Stacking operations**

- [: save in stack status of turtle which is current direction position and width of line
- ]: recover from stack status of turtle

To make a branch, the turtle must draw one part of the branch and then come back to the fork position and draw the other part of the branch. Part of a string can be saved for processing by putting it within brackets [ ].

Example, consider the rules

SET axiom X

X ==> g [ ~ + Y g ] g — within [ ]'s is a branch; the ~ means decrement the width of the line
Y ==> + Y

SET d 18
SET iniwidth 4
SET incwidth 1 — increment for changing width of line
SET angle 30
SET color black

First string in L is g[~+Yg]g. To draw this first draw the first line for the first g. At this point, save the [~+Yg] along with the current direction and the current width of the line. Continue drawing at the first symbol past the ]. Draw a line. Now that the end of the string has been reached, come back to the point in the drawing where the branch occured and draw the string within the [ ]'s.

What is L? L =

Suppose we change the X rule above by adding X onto the end:

X ==> g [ ~ + Y g ] g X

Now the L-system looks like:

**Example of drawing plants via L-systems**

Now we will examine some examples of growing plants.

**Example:**

This is in file: plant1

```
SET axiom X

X ==> g [ ~ + g Y ] g X
Y ==> g g [ ~ + g leaf ]
leaf ==> [ color Green {+ f - f f - f + % + f - f f - f} ]

SET d 18
SET iniwidth 4
SET incwidth 1
SET angle 18
SET color black
```

Differences from previous examples:

- The color field allows you to specify a color for part of the drawing.
- The parenthesis { } are used to define a region that is to be filled in.

**Example:**

This is in file: plant2

```
SET axiom X

X ==> g [ ~ + g Y ] [ ~ - - g Y ] g X
Y ==> X g g [ ~ + g leaf ]
leaf ==> [ color Green {+ f - f f - f + % + f - f f - f} ]

SET d 18
SET iniwidth 4
SET incwidth 1
SET angle 18
SET color black
```

**Fractals**

**References:**


- The Algorithmic Beauty of Plants, by P. Prusinkiewicz and A. Lindenmayer
- Automata, Languages, Development, by A. Lindenmayer and G. Rozenberg