Parallel Generation of L-Systems

GPUs at the service of simulation

Juan Rafael García Blanco

Facultad de Informática
Universidad Politécnica de Madrid

Complex Systems Simulation 2012

Table of Contents

Table of Contents

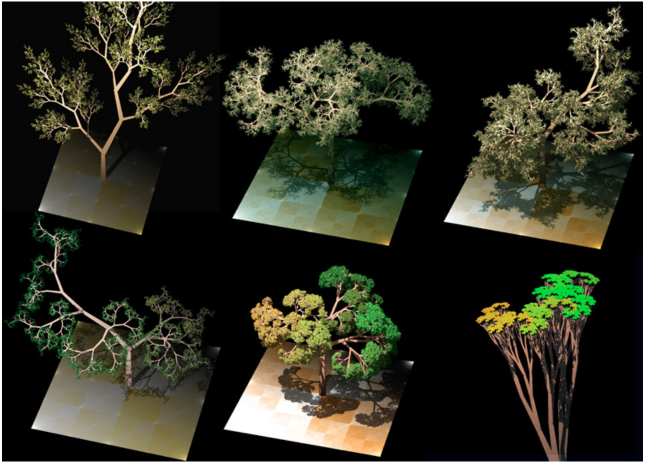
Origins of L-Systems

- Lindenmayer systems – or L-systems for short – are parallel rewriting systems.
- Introduced and developed in 1968 by the theoretical biologist and botanist A. Lindenmayer.
- Conceived as a mathematical theory of the development of simple multicellular organisms.
- Used to illustrate the neighborhood relationships between plant cells.

What Is an L-System? /1

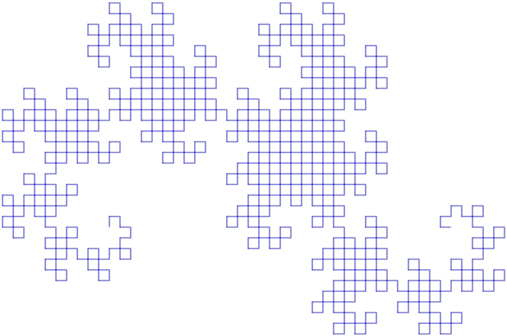
- The original system was extended.
- Now can be used to describe higher plants and complex branching structures.
- Versatile tool for plant modeling.
- Emphasis on plant topology, but also used to model the morphology of a variety of organisms.

What Is an L-System? /2



What Is an L-System? /3

- Can also be used to generate self-similar fractals, e.g. iterated function systems.
- Koch curve, Sierpiński triangle, Dragon curve, . . .



What Is an L-System? /4

An L-system consists of

- an alphabet of symbols.
- a collection of production rules.
- an initial axiom string.
- an interpretation mechanism.

Generation and Interpretation

- Rules are applied on variable symbols.
- A geometric structure is produced from the resulting string.

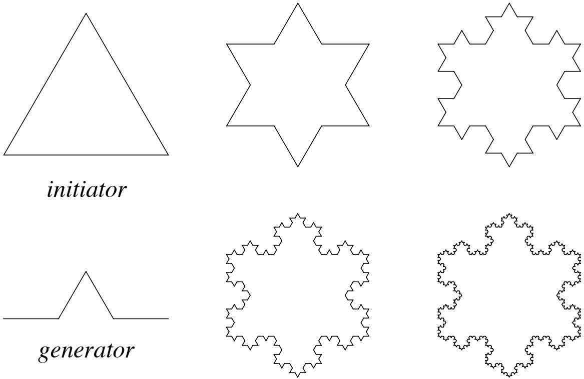
Table of Contents

General Formulation

- Rewriting means replacing subterms of a formula with other terms.
- Define complex objects by successively replacing parts of a simple initial object.
- Wide range of methods: polygons, strings, . . .
- Can be non-deterministic: set of applicable rules.
- Conway's *game of life* is an array-rewriting mechanism.

Koch Snowflake

- *Snowflake curve* construction given by Mandelbrot.
- "*...consists in replacing each straight interval with a copy of the generator ...*"



String Rewriting Systems (SRS) /1

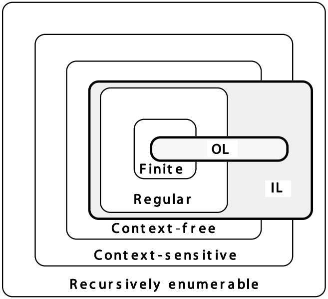
- Historically called semi-Thue systems.
- Rewriting system over strings from an alphabet.

Formal definition

- A semi-Thue system is a tuple (Σ, R).
- Σ is an alphabet, usually finite.
- R is a set of binary relations on strings: $R \subseteq \Sigma^\star \times \Sigma^\star$.
- The tuple $(u, v) \in R$ is called a rewriting rule or production rule and is usually written as $u \to v$.

String Rewriting Systems (SRS) /2

- Connection with Chomsky hierarchy.
- Chomsky applied the concept of string rewriting to describe the syntactic features of natural languages.
- Sets of strings and methods for generating, recognizing and transforming them.



String Rewriting Systems (SRS) /3

- L-systems are a type of SRS.
- In Chomsky grammars productions are applied sequentially.
- Whereas in L-systems they are applied in parallel.
- Intended to capture cell divisions in multicellular organisms.
- Great impact on the formal properties of SRS:
  - There are languages which can be generated by context-free L-systems,
  - But not by context-free Chomsky grammars.

Table of Contents

Main Idea /1

Informal definition

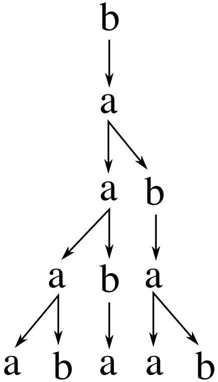
- Deterministic: exactly one production for each symbol.
- Context-free: rules application depends only on a single symbol.

Example

- Strings built of two letters a and b.
- Rules: $a \to ab$, $b \to a$.
- Axiom: just b.
- What happens after 4 iterations?

Main Idea /2

- Bear in mind that multiple rules are applied simultaneously.
- Resulting string is *abaab*.



Formal Definition /1

Components

- A string OL-System is an ordered triplet $G = \langle V, \omega, P \rangle$.
- V denotes an alphabet.
- $V^\star$ is the set of all words over V.
- V^+ is the set of all nonempty words over V.
- $\omega \in V^+$ is called the axiom.
- $P \subset V \times V^\star$ is a finite set of productions.

Formal Definition /2

Productions

- A production (a, χ) is written as $a \rightarrow \chi$.
- a is called the predecessor of the production.
- χ is called the successor of this production.
- For any $a \in V$ there exists a production $a \rightarrow \chi$; $a \rightarrow a$ productions are assumed for symbols with no other productions.
- DOL-Systems are those for which there is exactly one string χ associated with each symbol a.
  - If $a \rightarrow \chi \in P$ and $a \rightarrow \xi \in P$ then $\chi \equiv \xi$.
  - No ambiguity is possible.

Formal Definition /3

Derivation

- The string $\nu = \chi_1 \dots \chi_m \in V^\star$ is directly derived from $\mu = a_1 \dots a_m$ iff $a_i \rightarrow \chi_i$ for all $i = 1, \dots, m$.
- This is noted as $\mu \Rightarrow \nu$.
- μ_n is a derivation of length n by G iff the sequence $\mu_0 = \omega \Rightarrow \mu_1 \Rightarrow \dots \Rightarrow \mu_n$ can be created.

Example: Development of a Filament /1

- System used to simulate the development of a fragment of a multicellular filament such as that found in various algae.
- Symbols a and b represent cytological states of the cells (size, readiness to divide).
- Subscripts l and r indicate cell polarity, specifying the positions in which daughter cells will be produced.

Example: Development of a Filament /2

$G = \langle V, \omega, P \rangle$ definition

- $V = \{a_l, a_r, b_l, b_r\}$.
- $P = \{p_1, p_2, p_3, p_4\}$

$$\begin{aligned}
\omega &: a_r \\
p_1 &: a_r \rightarrow a_l b_r \\
p_2 &: a_l \rightarrow b_l a_r \\
p_3 &: b_r \rightarrow a_r \\
p_4 &: b_l \rightarrow a_l
\end{aligned}$$

Example: Development of a Filament /3

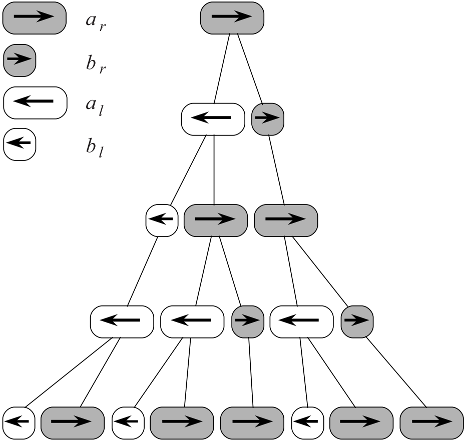


Table of Contents

Turtle Graphics /1

Current method

- Szilard and Quinton showed that strikingly simple DOL-systems could generate fractals.
- These results were subsequently extended in several directions.
- Realistic modeling of herbaceous plants.

Turtle

- A state of the turtle is defined as a triplet (x, y, α).
- (x, y) represents the turtle's position.
- α, called heading, represents the direction in which the turtle is facing.

Turtle Graphics /2

- Each symbol of V can be assigned a command.
- The turtle can respond to these commands.
- Step size d and angle increment δ.

Typical commands

F Move forward a step of length d. The state changes to $(x + d\cos\alpha, y + d\sin\alpha, \alpha)$.

$+$ Turn left by angle δ. The state changes to $(x, y, \alpha + \delta)$.

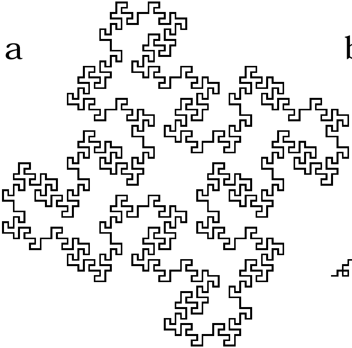
- Turn right by angle δ. The state changes to $(x, y, \alpha - \delta)$.

Turtle Graphics /3

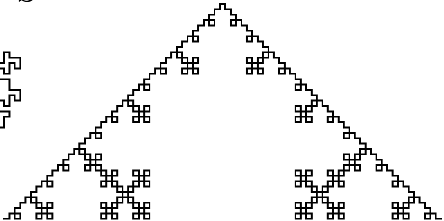
- Given
  - a derivation ν.
  - an initial state (x_0, y_0, α_0).
  - fixed parameters d and δ.
- The turtle interpretation of ν is the figure drawn by the turtle in response to the string ν.
- Sequential interpretation: one symbol after another.

Turtle Graphics: Koch Island and Koch Quadratic Curve

a



b

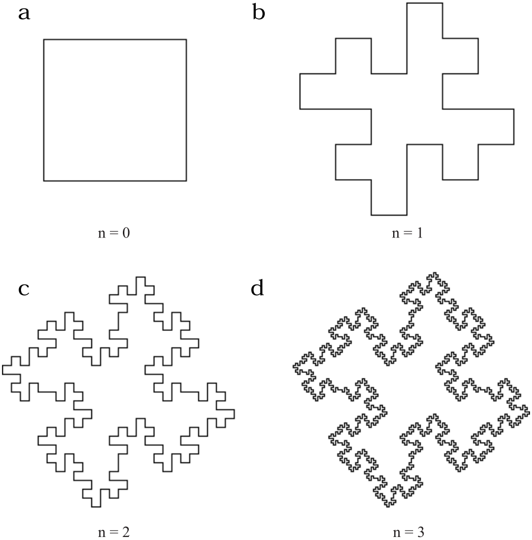


$n = 2, \ \delta = 90^\circ$
F-F-F-F
$F \rightarrow$ F+FF-FF-F-F+F+F

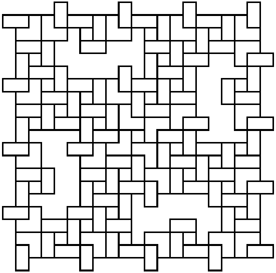
F-F-F+F+FF+FF-F

$n = 4, \ \delta = 90^\circ$
-F
$F \rightarrow$ F+F-F-F+F

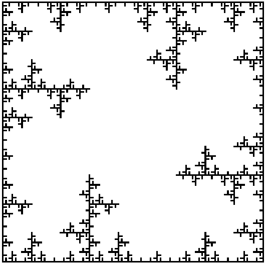
Turtle Graphics: Koch Island



Turtle Graphics: More Curves

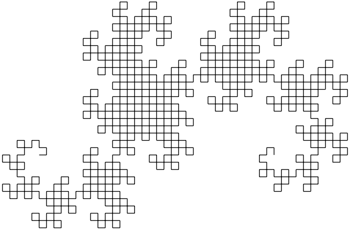


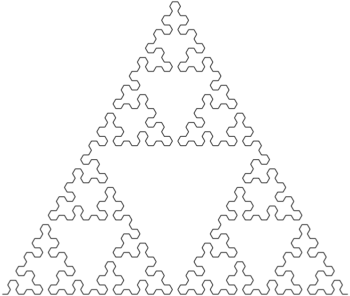
c $n = 3, \ \delta = 90^\circ$
F-F-F-F
F $\rightarrow$ FF-F+F-F-FF



d $n = 4, \ \delta = 90^\circ$
F-F-F-F
F $\rightarrow$ FF-F--F-F

Turtle Graphics: Dragon Curve and Sierpiński Triangle





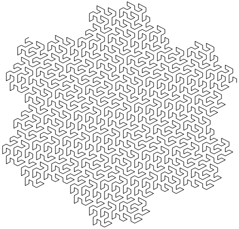
a n=10, δ=90°
F_l
$F_l \rightarrow F_l + F_r +$
$F_r \rightarrow - F_l - F_r$

b n=6, δ=60°
F_r
$F_l \rightarrow F_r + F_l + F_r$
$F_r \rightarrow F_l - F_r - F_l$

Table of Contents

Edge Rewriting

- Can be viewed as an extension of Koch constructions.
- Two types of edges, "left" and "right", F_l and F_r.
- space-Filling, self-Avoiding, Simple and self-Similar curves (FASS).



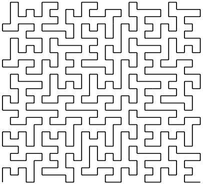
a

n=4, δ=60°

F_l

$F_l \rightarrow F_l+F_r++F_r-F_l--F_lF_l-F_r+$

$F_r \rightarrow -F_l+F_rF_r++F_r+F_l--F_l-F_r$



b

n=2, δ=90°

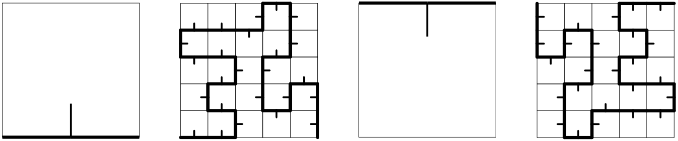
$-F_r$

$F_l \rightarrow F_lF_l-F_r-F_r+F_l+F_l-F_r-F_rF_l+$
$F_r+F_lF_lF_r-F_l+F_r+F_lF_l+$
$F_r-F_lF_r-F_r-F_l+F_l+F_rF_r-$

$F_r \rightarrow +F_lF_l-F_r-F_r+F_l+F_lF_r+F_l-$
$F_rF_r-F_l-F_r+F_lF_rF_r-F_l-$
$F_rF_l+F_l+F_r-F_r-F_l+F_l+F_rF_r$

Space-Filling, Self-Avoiding, Simple and Self-Similar Curves

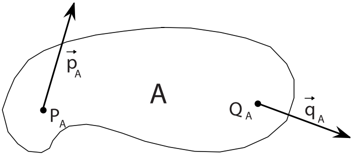
- Self-avoiding approximations of curves that pass through all points of a square.
- Algorithms exploit the relationship between such a curve and a recursive subdivision of a square into tiles.
- F_l and F_r are replaced by polygons.
- Recursive application indicates that the whole curve is approximately space-filling.



$$
\begin{aligned}
\mathtt{F_l} \rightarrow\ & \mathtt{F_lF_l{+}F_r{+}F_r{-}F_l{-}F_l{+}F_r{+}F_rF_l{-}F_r{-}F_lF_lF_r{+}} \\
& \mathtt{F_l{-}F_r{-}F_lF_l{-}F_r{+}F_lF_r{+}F_r{+}F_l{-}F_l{-}F_rF_r{+}} \\
\mathtt{F_r} \rightarrow\ & \mathtt{{-}F_lF_l{+}F_r{+}F_r{-}F_l{-}F_lF_r{-}F_l{+}F_rF_r{+}F_l{+}F_r{-}} \\
& \mathtt{F_lF_rF_r{+}F_l{+}F_rF_l{-}F_l{-}F_r{+}F_r{+}F_l{-}F_l{-}F_rF_r}
\end{aligned}
$$

Node Rewriting

- Turtle interpretation is extended by symbols which represent arbitrary subfigures.
- Each subfigure A is represented by
  - two contact points: entry point P_A and exit point Q_A.
  - two direction vectors: entry vector $\vec{p}_A$ and exit vector $\vec{q}_A$.
- When incorporated, A is aligned with the current position of the turtle.
- The turtle is assigned the exit parameters of A.



Node Rewriting: Example

- Rewriting rules:

$$L_{n+1} = +R_nF - L_nFL_n - FR_n+$$
$$R_{n+1} = -L_nF + R_nFR_n + FL_n-$$

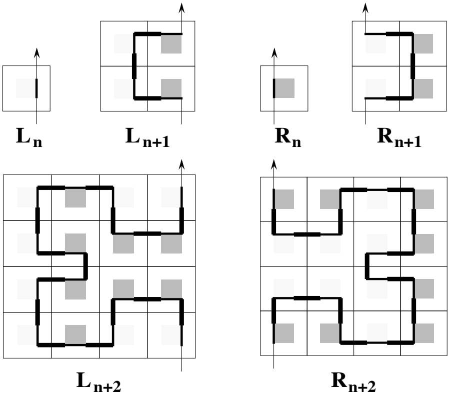
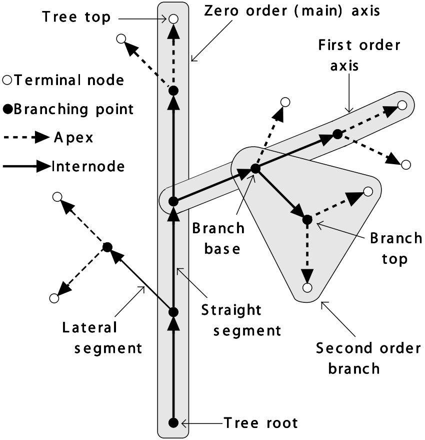


Table of Contents

Axial Trees

- The plant kingdom is dominated by branching structures.
- An axial tree is a special type of rooted tree.



Tree OL-Systems

Components

- V denotes a set of edge labels.
- $\omega \in V^+$ is an initial tree.
- $P \subset V \times V^\star$ is a finite set of productions.

Construction

- A tree production replaces a predecessor edge by a successor axial tree.
- The starting node of the predecessor is identified with the successor's base.
- And the ending node is identified with the successor's top.

Bracketed OL-Systems

New structure

- Axial trees can be represented using strings and brackets.
- Brackets are used to delimit a branch.
- Two new operations are introduced.

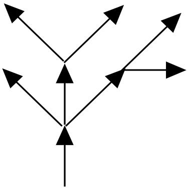
New operations

[Push the current state of the turtle.

] Pop the top state from the stack and make it the current state of the turtle.

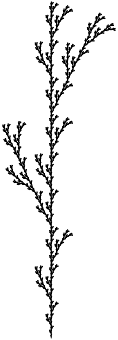
Bracketed OL-Systems: Example

- Derivation proceeds as in OL-systems without brackets.
- Brackets replace themselves.

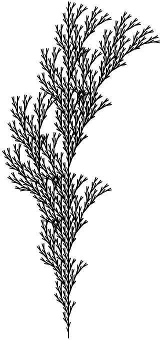


F[+F][-F[-F]F]F[+F][-F]

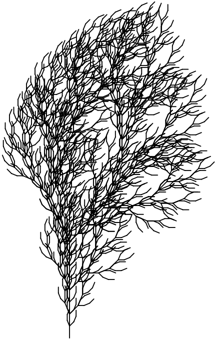
Bracketed OL-Systems: 2D Trees /1



a
n=5, δ=25.7°
F
F → F[+F]F[-F]F

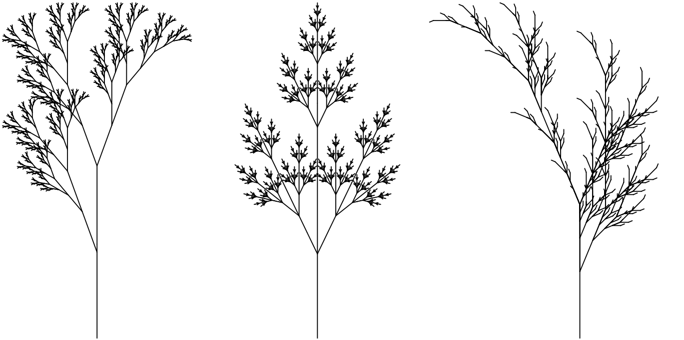


b
n=5, δ=20°
F
F → F[+F]F[-F][F]



c
n=4, δ=22.5°
F
F → FF-[-F+F+F]+
[+F-F-F]

Bracketed OL-Systems: 2D Trees /2



d

n=7, δ=20°

X

X →F[+X]F[-X]+X

F →FF

e

n=7, δ=25.7°

X

X →F[+X][-X]FX

F →FF

f

n=5, δ=22.5°

X

X→F-[[X]+X]+F[+FX]-X

F→FF

Table of Contents

Stochastic L-Systems

- Combining plants generated by the same L-system would produce an artificial picture.
- Solution: introduce variations.
- Preserve the general aspects of a plant but modify its details.

Formal definition

- A stochastic L-system is an ordered quadruplet $G_\pi = \langle V, \omega, P, \pi \rangle$.
- Function $\pi : P \rightarrow (0, 1]$ maps the set of productions into the set of production probabilities.
- The sum of probabilities of all productions with the same predecessor is equal to 1.

Context-Sensitive L-Systems

- Production application also depends on the predecessor's context.
- Useful in simulating interactions between plant parts.
- In a (k,l)-system the left context is a string of length k and the right context is a string of length l.
- $a_l < a > a_r \to \chi$ is a production of a 2L-system.

Parametric L-Systems

- It is difficult to capture continuous phenomena.
- Solution: numerical parameters associated with L-system symbols.
- Strings of modules consisting of letters with associated parameters.

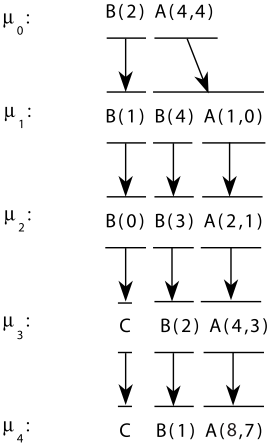


Table of Contents

Graphics Processors (GPUs) and OpenCL

- Massively parallel architecture.
- Thousands of threads executing concurrently.
- Relaxed memory consistency model.
- *Kernels* are executed in the GPU side.
- OpenCL-C code is portable.
- OpenGL Vertex Buffer Objects can be modified from OpenCL *kernels*.
- Parallel derivation and interpretation: everything is done in the GPU side.

Problem Statement

- Input: number of iterations and angle δ.
- Output: graphical representation of the L-system.
- Axiom and system definition need to be transferred to GPU memory.
- Nothing needs to be retrieved from GPU memory.
- Focus on DOL-systems.

Table of Contents

Representation

- Each symbol (letter) is assigned an integer number.
- Rules stored in texture memory for efficient accesses.
- Each production rule occupies one row of a 2D matrix.
- First position stores the amount of symbols that are in the successor of the rule.

2D image

4	F	B	+	A
1	+			
1	-			

⋮

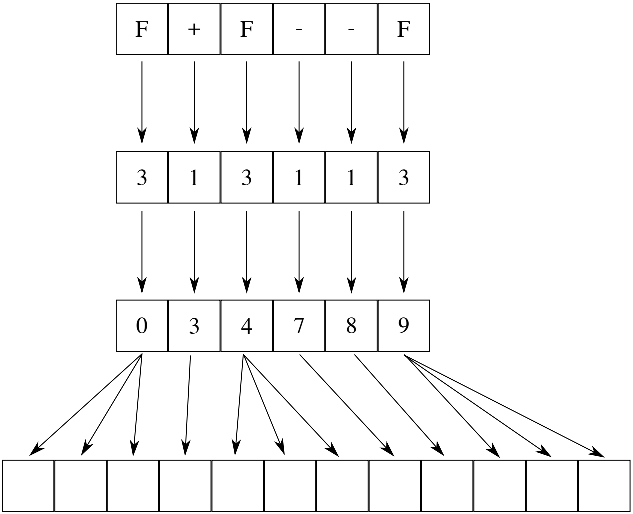
$$F \rightarrow FB + A$$

$$+ \rightarrow +$$

$$- \rightarrow -$$

$$\dots$$

Derivation Passes /1



Derivation Passes /2

Algorithm

- Each symbol is assigned one or multiple threads.
- First, each thread computes the amount of space needed for their associated symbol.
- Second, a sum-scan is performed over the whole buffer.
- Third, production rules are applied.

Performance considerations

- Pass 1 is very efficient.
- Efficient parallel sum-scan algorithms exist for GPUs.
- Pass 3 is highly inefficient because of non-contiguous accesses.

Table of Contents

Representation

- Turtle state can be modified by 3x3 matrices: “move forward”, “turn right”, “turn left”, . . . ; but not push and pop.
- Matrices associated with each symbol are stored in global memory.
- In front of each matrix the amount of geometry is stored.
- An OpenGL Vertex Buffer Object is allocated to hold the resulting points.

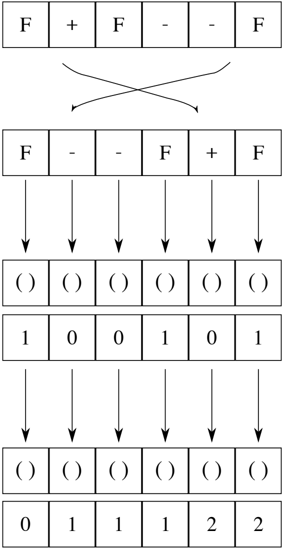
move forward matrix

1	1	0	0	0	1	0	1	0	1

turn left matrix

0	0	1	0	-1	0	0	0	0	1

Interpretation Passes /1



Interpretation Passes /2

Algorithm

- Multiplication in local axis.
- Matrices are placed in reverse order.
- A multiply-scan is performed over the matrices using left multiplication.
- A sum-scan is performed over the buffer that holds the point counters.

Performance considerations

- Performance depends on matrix multiplication.
- Not very efficient.

Demo

Table of Contents

Summary

- L-systems are capable of modeling a wide range of natural phenomena and fractals.
- They are inherently parallel systems.
- GPUs could improve the performance of the derivation step.
- Complex algorithms could be developed to improve the interpretation step.

Thank you

Bibliography I

Ron Goldman, Scott Schaefer, and Tao Ju.
Turtle geometry in computer graphics and computer-aided design.
Computer-aided Design, 36:1471–1482, 2004.

Markus Lipp, Peter Wonka, and Michael Wimmer.
Parallel generation of multiple L-systems.
Computers & Graphics, 34(5):585 – 593, 2010.

P. Prusinkiewicz, A. Lindenmayer, and J. Hanan.
The algorithmic beauty of plants.
Virtual laboratory. Springer-Verlag, 1990.