ソフトウェア工学 105－9
（1995．9．8）

GUIを用いたL-systemエディタの開発

西村 剛* 有澤 誠**
gon@sfc.keio.ac.jp
*慶應義塾大学大学院 政策・メディア研究科
**慶應義塾大学 環境情報学部
〒242藤沢市遠藤5322

我々は形式言語の一種である，L-systemの生成規則及びパラメータを決定し，モデルを構築するための支援をおこなうエディタを開発した．このエディタはGraphicalUserInterfaceを用いて，インタラクティブにパラメータの変更をおこなうことができる．本エディタは，生成規則から図形を描画することに重点をおくのではなく，図形をもとに，生成規則をつくる為の支援をおこなうことに重点をおいて開発された．本稿は，L-systemを用いて，生成規則をより直観的に，容易に作成するための提案を行う．

The development of L-system editor using the Graphical User Interface.

Go Nishimura* and Makoto Arisawa**
*Graduate School of Media and Governance, Keio Univ.
**Faculty of Environmental Information, Keio Univ.
5322, Endo, Fujisawa-shi, 242, Japan

We have developed an editor that decides the rule and the parameter of the L-system, type of formal language, and helps structurize the model. This editor can change a parameter interactively using Graphical User Interface. This editor was developed as supporter for production rule making, not as visualized tool. The paper suggests the way that to make the production rule intuitionally.

1.はじめに

1.1 L-system

L-systemは，1968年にLindenmayerが提唱した，形式言語の一種である[1][2]．各アルファベット（シンボル）にタートルの動きを割り当てる事によって3次元構造を生成し，植物をはじめとする，視覚的に生体に似た構造を表現することができる．

L-systemの特徴の一つには，終端記号がないことがあげられる．よってシステムの定常的な状態は，同じアルファベットに書き換えられる状態として表現される．

書き換え規則の形態，書き換えが決定論的に行われるか否か，等の違いにより，DOL,IL,OL,MAP-L-systemといった種類に大別される[3]．また，各シンボル毎にパラメータを付けるParametricL-systemと，パラメータを持たない，Non-parametric L-systemがある．

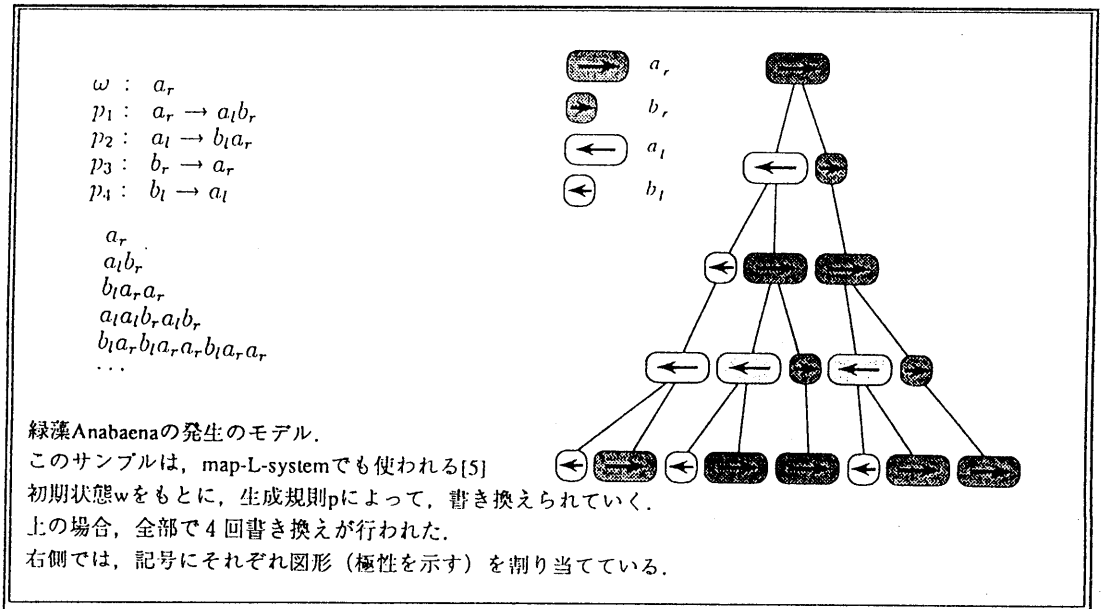
ParametricL-systemでは，タートルの回転や，前進に割り当てられているシンボルに対して，個別にパラメータを指定することにより，回転角度や，前進する距離をシンボル毎に決定する．Non-parametricL-systemは，単純な生成規則で済むかわりに，回転角度や前進する距離が，どのシンボルに対しても等しくなる．

図1Aは，もっとも単純なL-systemであるDOL-systemによる書き換えの例である．

リンデンマイヤーのL-systemは，記号列に対して書き換えが行われる.これに対し，L-systemをグラフに拡張し，要素間の関係を記述できるようにしたグラフ発生システム[5][10]もある．

いくつかの生成規則から，再帰的にアルファベットの置き換えが行なわれるこのモデルでは，パラメータのわずかな違いが構造に大きな影響を及ぼす．また，ルールやパラメータを一見しただけでは，どのような構造なのか直観的にわかりづらい．

コンピュータを用いて，与えられた生成規則，初期条件，視点等をもとに，書き換えを行い，タートルグラフィックスをグラフィックディスプレイに描画するアプリケーションや，



緑藻Anabaenaの発生のモデル．
このサンプルは，map-L-systemでも使われる[5]
初期状態wをもとに，生成規則pによって，書き換えられていく．
上の場合，全部で4回書き換えが行われた．
右側では，記号にそれぞれ図形（極性を示す）を割り当てている．

図1A:DOL-system[3]

レンダリングアプリケーション用のスクリプトを出力するアプリケーションが，すでにいくつか開発されている．しかし，どのアプリケーションも，モデルを植物に限定したり，あるいは，生成されたモデルを観察することに重点がおかれている．生成規則をもとにグラフィックスを得る，という発想とは逆に，グラフィックスをもとに，より好ましい生成規則を得る，という発想も大切ではないだろうか．

そこで，今回我々はL-systemの生成規則及びパラメータを決定し，モデルを構築する為の支援をGUIを用いて行なうL-systemエディタ*ledit*（以下*ledit*）を開発した．

1.2 *ledit*概要

*ledit*の画面を図1Bに示す．*ledit*は，UNIXワークステイション上でC言語＋X-motif（約4000ステップ）を使用して開発した．ウィンドウ全体のサイズは，960x650ピクセル，モデル表示部分は，500x500のピクセルを表示する．また，この表示部分をそのままpostscript形式のファイルに変換し，印刷することが可能である．

*ledit*は、次の3つの特徴を持っている．

- モデルを3D視覚化し、自由な角度からモデルを観察することを可能にする。
- インタラクティブにパラメータを変更し、モデルに及ぼす影響を視覚的に分析することを可能にする。
- 視覚化されたモデルをもとに、画面内で選択された特定の部分に対し、どのルールが影響を及ぼしているか観察できる。また、その場でルールの変更・追加を行なうことを可能にする。

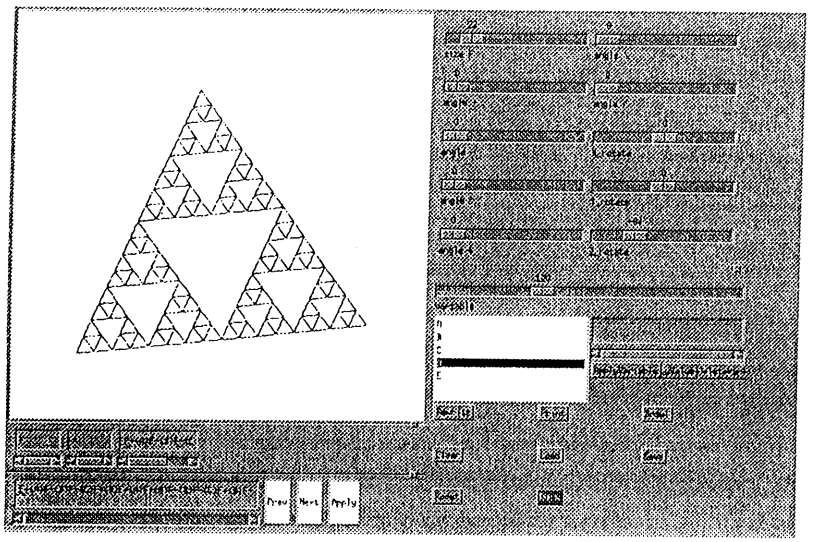


図1B:*ledit*全体画面

また，このエディタでは、特に以下の工夫を行なった．

- 各アルファベットの書き換えの履歴を保存．
- スケールを用いたパラメータ変更機能を実装．
- タートルグラフィックスをそのまま画面に描画せずに、始点終点をメモリに保存した後表示．

*ledit*では，第一段階として，DOL-system及び，OL-systemを対象とした．

2.システム構成

*ledit*は，文字列置換，書き換え履歴保存，数値演算，変数管理，３次元タートル描画，画面管理，生成規則検索，postscript印刷の7つの部分からなる（図2A）．
parametricL-systemに対応するために，数値演算部と変数管理部を別に設け，パラメータの処理を一括して行うことにした．

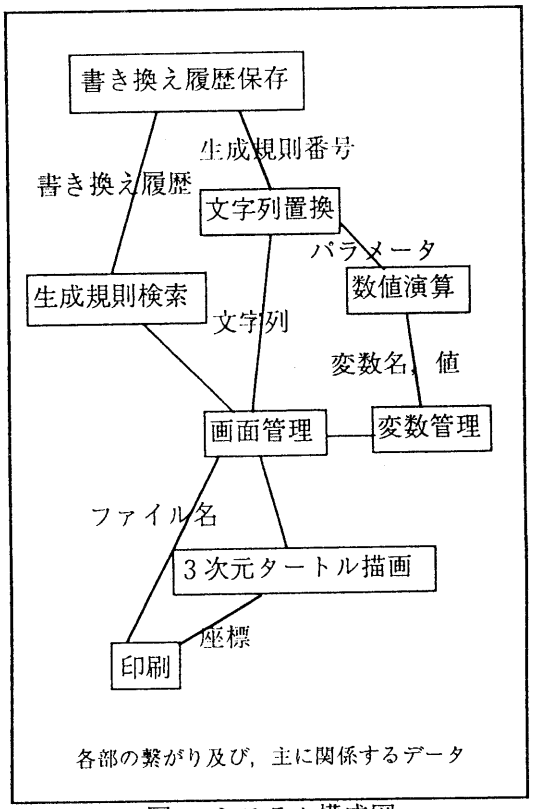


図2A:システム構成図

2.1　文字列置換

*ledit*では，書き換え後の記号列を画面に表示する．書き換えは左から順にアルファベット単位で行われ，一文字書き換えを行う度に，書き換え後の長さ分メモリを確保して，char型の配列で記号列を記憶している．
L-systemでは，通常書き換え規則とともに何回書き換えを行うか指定する．単純に規則を書き換えるだけなら，再帰的に書き換えをおこなったほうが，計算量が少なくてすむ．しかし，アルファベット毎に再帰的に書き換えを行う方法では，context-sensitiveな生成規則の書き換えは行えない．これは，context-sensitiveな生成規則では，一世代前の左右両隣のアルファベットを参照して，書き換えがおこなわれる可能性があるためである．

2.2　書き換え履歴保存，生成規則検索

生成規則の数が合計で16個未満のときは，表示部に描画されたモデルをもとに，同じ書き換え過程を経て生成された部分を検索することが可能である．この機能を実現するために，記号列と同じ長さを持つint型の配列を用意し，そこに書き換え履歴として，書き換えの際に用いられた生成規則の番号を保存することにした．書き換える前の変数を，int*before，書き換えた後の変数をint*afterとすると，*afterは次のように表される．

*after = *before << 4 + production_rule_number;

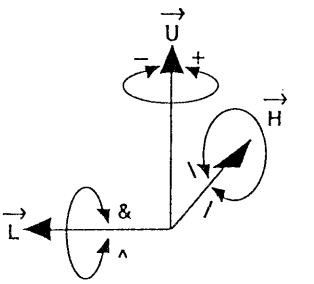
書き換え前の変数を４ビット左にシフトさせた後に，生成規則の番号を足している．あらかじめ各生成規則の番号を，0～15と決めておくことにより，過去8回(32Bit / 4bit)の書き換えに用いられた生成規則までを記録することができる．
この配列変数は，記号が書き換えられる度に，書き換えられた記号列の長さと同じ数だけ順に追加して記録される．記号列の先頭からみた各記号の位置と，書き換え履歴用配列の先頭からみた，その記号の書き換え履歴の位置は同じであり，一対一に対応している．よって，この書き換え履歴の値が等しいか否か，で２つの記号が同じ生成規則から生成されたどうかを判断することができる．

3.2　数値演算，変数管理

文字列計算ライブラリstrvalue及びvariable1を実数演算対応にして使用した．この文字列計算ライブラリは，四則演算，論理演算，変数の宣言，変数の値の管理を行う．Non-parametricL-systemでパラメータが変数の場合には，書き換えが行われる前にあらかじめこのライブラリを用いて変数として宣言される．

3.3　３次元タートル描画

drawボタンの選択，もしくはパラメータスケールの変更アクションが生じた場合，ウィンドウ上の表示部にタートルグラフィックスが描かれる．
はじめに生成された記号列をもとに，３次元でタートルグラフィックスの各線分の頂点の世界座標を計算する．記号列の中のシンボルには，タートルの動きが割り当てられている（図3A)．各頂点の世界座標は，他のデータとともにリスト構造で保存される．その後，チェインをたどり，各頂点を視点変更スケールで指定されたローカル座標に変換し，最終的に画面管理部から指定された始発点をもとに，表示部に描画される．
視点変更スケールを変更した場合には，最後のローカル座標に変更する計算だけを繰り返せばよいため，多数の線分からなるグラフィックスも敏速に視点を変更し，観察することが可能になる．



シンボル	タートルの動き
F	Move forward
+	Turn left
-	Trun right
&	Pitch down
^	Pitch up
¥	Roll left
/	Roll right
¦	Trun around

図3A:シンボル
(一部) [3]

4．ユーザインタフェース

ユーザインタフェースの観点からみた本エディタの特徴として，

- スケールを用いたパラメータ変更機能
- 表示部をクリックすることにより生成規則を選択する機能
- 生成規則及びパラメータの変更を瞬時に視覚化する機能

の３つをあげる．

4.2. スケールを用いたパラメータ変更機能

スケールは，画面右側にまとめてあり，パラメータの変更を行うことができる．全部で11個のスケールが配置されているが，それぞれは次の3種に大別される．

- non-parametric L-systemで角度を変更する際に用いるスケール
  +,-,&,^,¥,/
- parametric L-systemで，変数の値を変更する際に用いるスケール
- 表示画像の大きさ，視点の位置を変更する際に用いるスケール
  F, X-rotate,Y-rotate, Z-rotate

4.2.1 Non-parametric L-system用スケール

non-parametric L-systemは，パラメータを持たない．各シンボルのパラメータを設定せず，すべて同じ値としてあらかじめ別に設定する．このため，記号列中で，パラメータである数値と書き換えの対象となるシンボルとを区別する必要がない．Non-parametric L-systemを用いることによって，コッホ曲線や，シェルピンスキーのガスケットを描くことができる．しかし，各シンボルのパラメータがすべて等しいという制約から，描くことのできる図形もまた大きく制限される．そこで，*ledit*では，Non-parametric L-systemの生成規則に対しても，あえて各シンボル毎にパラメータを設定できるようにした．ユーザは，生成規則にパラメータを入れなくとも，各シンボルに対応したスケールの値を変更することによって，これまでのNon-parametric L-systemより幅の広いモデルを構築することが可能である．

4.2.2 Parametric L-system用のスケール

Parametric L-systemでは，シンボルの前に括弧を用いてパラメータを指定することができる．このパラメータは，タートルグラフィックスを描く時に参照され，曲がる角度や，進む距離を決定する．パラメータには，数字を入れることも変数を入れることも可能である．

*ledit*では，変数リストで選択した変数の値を，スケールを用いて変更することが可能である．テキストフィールドに，使いたい変数名を入力することで，以降，その変数はリストに登録される．よって，あらかじめ生成規則の任意のシンボルの前にパラメータとして変数名を埋め込み，変数リストに，使用する変数名を登録することで，変数に代入する値をスケールを用いて決定することが可能となる．

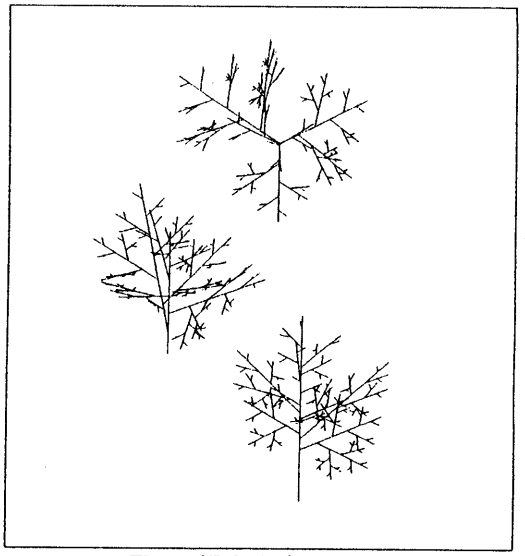


図4A:視点を変更して木を観察

4.2.3 視点変更用のスケール

視点変更用のスケールには，拡大縮小と，角度変更の2種類がある．拡大縮小のスケールは，タートルが一度に進む距離を変更する．角度変更のスケールX-rotate,Y-rotate,Z-rotateは，それ

ぞれ，ローカル座標の各軸の向きを変更する．スケールが選択されると，自動的に世界座標からローカル座標への変換がおこなわれ，表示部にモデルが描画される．

5. 評価

*ledit*を実際に使用してみると、パラメータ変数をスケールで変更し，インタラクティブにタートルグラフィックスを観察できる機能が，非常に効果的であることがわかる．スケールを用いて変数の値を変更すると，表示部に描画されるタートルグラフィックスのモデルが，リアルタイムに，そして動的に変化する(図4B)．ユーザは，どの生成規則のどのシンボルのパラメータを何度変更して，という細かいことは意識せずに済み，直観的にスケールを左右に動かすだけで，パラメータの変化がおよぼす影響をすぐに観察できる．

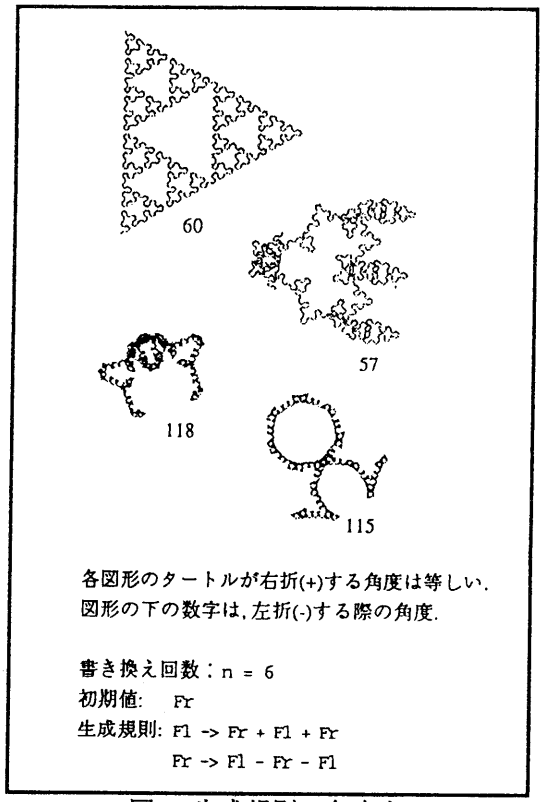


図4B:生成規則の角度を変更した例

モデルの一部分の構造を集中的に変化させたい場合には，まず表示部のポップアップメニューから，生成規則参照モードを選択する．次にタートルグラフィックスの希望する線分をクリックすれば，その線分のもととなったシンボルがどの生成規則で書き換えられ，描画されたかがわかる．その後，参照された生成規則に用いられているパラメータをスケールで変更すればよい．

生成規則や初期値，使用する変数名の宣言，変更，削除以外のすべての動作は，マウスだけでおこなえ，直観的にモデルを観察することを可能にした．

5. 今後

*ledit*のコンセプトの一つに，「生成規則から，画像出力を得る従来のアプリケーションとは逆の方向性，すなわち，画像から，生成規則をつくる支援をおこなう」，という発想がある．現バージョンでは，生成されたタートルグラフィックスから，もとの生成規則及び，パラメータを変更するにとどまっている．将来的には，何もない状態から，GUIを用いて，生成規則をつくる支援をおこなう機能をつけ加えたい．

今回は，特定の用途に焦点を絞った機能は実装していない．今後は，木構造のモデルにおいて，枝分かれの角度が最適となるよう，パラメータの値を自動的に計算する機能や，生成されたタートルグラフィックスのフラクタル次元を測定する機能をつけ加え，生体の構造を記述するための支援になるよう改良を加えていく予定である．

6. 謝辞

文字列計算ライブラリを提供して下さり，また，leditの開発において数々の助言を頂いた山川総司氏に深く謝意を表する．

参考文献

[1] A. Lindenmayer, Mathematical models for cellular interaction in development, Parts I and II, *Journal of Theoretical Biology*, 18:280, 1968.
[2] A. Lindenmayer, P. Prusinkiewicz, Developmental models of multicellular organisms: A computer graphics perspective, in C.G. Langton(ed.), *Artificial Life*, Addison-Wesley, 1989.
[3] A. Lindenmayer, Developmental systems without celluar interactions, their languages and grammars, *Journal of Theoretical Biology*, 30:455,1971.
[4] A. R. Smith, Plants, fractals, and formal languages, Proceeding of SIGGRAPH '85, Computer Graphics, 19(3):313, 1985.
[5] H. Doi, Graph-theoretical analysis of cleavage pattern: Graph developmental system and its application to cleavage pattern of ascidian egg, Develop. Growth and Differ., 26:49, 1984.
[6] H. Honda, Description of the form of trees by the parameters of the tree-like body, Effects of the branching angle and the branch length on the shape of the tree-like body, *Journal of Theoretical Biology*, 31:331, 1971.
[7] P. Prusinkiewicz, M. James, R, Mech,Synthetic Topiary, Computer Graphics Proceedings, Annual Conference Sereis, 351, 1994.
[8] P. Prusinkiewicz, J. Hanan, Visualization of botanical structures and processes using parametric L-systems, D. Thalmann, editor, *Scientific Visualization and Graphics Simulation*, 183, J. Wiley&Sons, Chichester, 1990.
[9] P. Prusinkiewicz, A. Lindenmayer, The algorithmic beauty of plants, springer verlag, 1990.
[10] 人工生命研究会編, 人工生命, 情報と生命とCGの交差点, 共立出版株式会社, 19
[11] 土井洋文, 生物のかたちづくり, 発生からバイオコンピュータまで, サイエンス社, 1988.
[12] 北野宏明, 進化するコンピュータ, 遺伝的アルゴリズムから人工生命へ, ジャストシステム,1993.

[1]慶應義塾大学大学院政策・メディア研究科修士１年　山川総司氏開発