20 A Configuration-Generating Method Based on A Lattice System

Chen, Hou-I and Liou, Shuenn-Ren
Institute of Building and Planing, National Taiwan University
Department of Architecture, Tung Hai University

In this paper, I would propose a configuration-generating method: "L grammars". Relating to Turing machine, cellular automata, tiling problem and Wang tile theory. It is named Lgrammars ″ .

L grammars base on an infinite lattice space (usually a chess-board-like lattice plane) is a bottom-up approach, in which each cell of a lattice system corresponds to a bounded configuration unit. Thus, a lattice plane would corresponds to a configuration union. The power of L grammars is demonstrated by generating various configurations such as fractal images (two dimensional examples) and crystal models (three dimensional examples).

Finally, It would be discussed briefly about some interesting issues concerning about L grammars such as the limitations of L grammars.

KEYWORDS: configuration, form, pattern, shape grammars, tiling, generation

1.0 INTRODUCTION

Shape grammars, analogizing to N. Chomsky's phrase structure grammars, is a powerful analysis and generation system. In terms of terseness and facility195, each grammar is different and unclear in details. and still has problems to be executed by computer. There might have other ways to be more specific to generate configurations.

Reviewing Turing machine, Wang Tile, and cellular automata, this paper would propose a generation method which bases at lattice system is a bottom-up approach. This generation method is named *L grammars*[196] for the first letter of *lattice.*

1.1 Generateing configurations by computation

The idea of L grammars is to ask is it possible to generate (two dimensional) configurations by computing? After a series of computation, Turing machine would left one dimensional data on a tape. If the erase function is instead of writing into another tape

which might be seen as the next time step of previous tape, there would have a two dimensional configuration. For example (figure 1,2), there is a Turing machine which erases the continuous integer "1" on the tape, and stopped when meeting "0". If the change of data was remain by every time step tape, we would have a right angled triangle (figure 2).

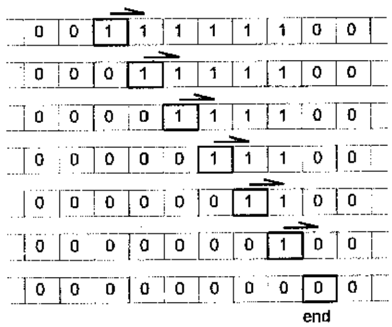


figure 1

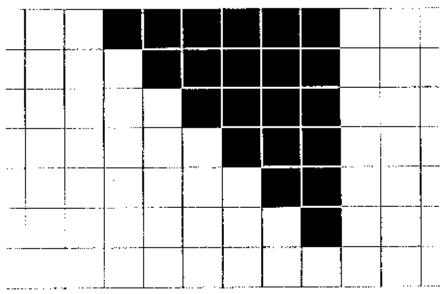


figure 2

Thus, the algorithm of generating the right angled triangle is identity to the Turing machine which erase continuous "1" on the tape. Now we had a simple idea which might generates two dimensional configuration by computing. In fact, the idea of time step is similar to cellular automata, configurations such as figure 2 is usually called space-time pattern. If we remark the state of Turing machine on the tape, and it would similar to Wang tile introduced by mathematician Hao Wang.

Wang tile, defined by Chinese mathematician Hao Wang, is a finite set of square tiles; the tiles are all the same size but each edge of a tile has a stipulated color and the colors are combined in several specified ways. We assume that we have infinitely many copies of each type of domino, and that we are not permitted to rotate[197] a tile in two dimension. The object of the tiling is to cover an infinite plane with tiles in such a way that adjacent edges have the same color. (Wang,1965:103-104) (figure 3)

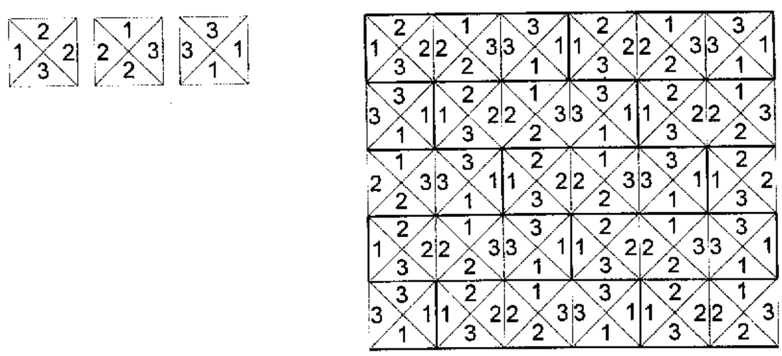


figure 3 colors are exchanged for numbers

The significance in theory stems from the fact which was pointed out by Wang that it is possible to find sets of Wang tiles which simulate the behavior of any Turing machine(Grunbaum & Shephard,1987: 584 & 604). That is, we could computing by tiles.

According to Wang tiles, I suggest a set of tiles which, defined by the continuity of the pattern of adjacent tiles, could calculate the highest common factor of two given positive integers....Let $S1$ to be the larger one of a and b , and $S2$ to be the smaller one. Let $d=S1-S2$, if $d>S2$, then let $d=d-S2$, until $d<S2$, let $S1=S2$ and $S2=d$, Now $S1$ and $S2$ would be seen as the new given positive integers and recalculated until $d=S2$ then d or $S2$ would be the answer—the highest common factor of a and b.

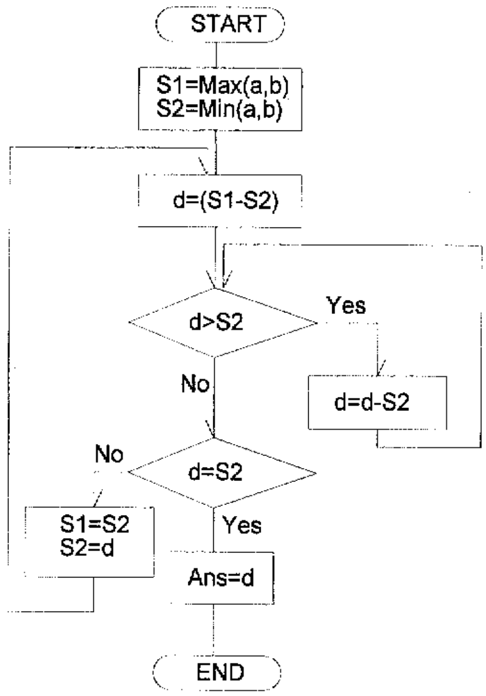
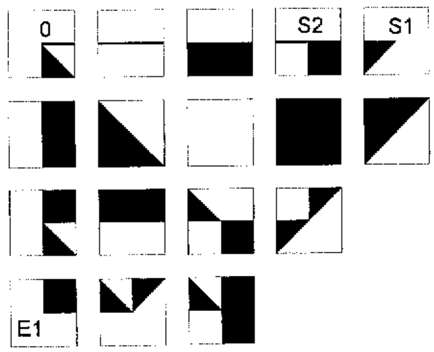


figure 4flowchart of finding the highest common factor of two numbers

For example, let $a=5$ and $b=2$, so $S1=5$ and $S2=2$, $d=S1-S2=5-2=3$. Because $d>S2$ (3>2) , $d=d-S2=3-2=1$. Now $d<S2$ (1<2) so let $S1=2$ and $S2=1$. 2-1=1, 1=1, the highest common factor of 5 and 2 would be 1.



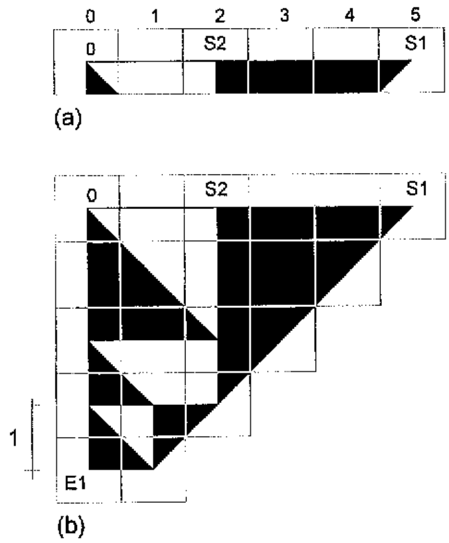


figure 5tiles for finding the highest common factor of two numbers *figure 6*

The set of tiles in figure 5 could get the highest common factor of *a* and *b*, Let *S1* to be the larger one of *a* and *b* , and *S2* to be the smaller one. tile  was put at the right side of tile  as the value of *a, b*. For example, let *a*=5 and *b*=2, so *S1*=5 and *S2*=2, the initial state of tiling show as figure 6(a). According to the same algorithm, there would have unique result of tiling as figure 6(b). The distance of tile  and tile  would be the output. In this case was 1.

2.1 THE DEFINITION OF L GRAMMARS

2.1 Two dimensional L grammars

With reference to Wolfram's study about cellular automata, I define two dimensional L grammars has four components:

1. An n-dimensional space of lattice system.
2. A finite set of symbols.
3. A finite set of rules.
4. A mapping list of configurations and symbols.

The plane of lattice system is the plane tiled or constructed by the parallelogram periodically, where the plane is the infinite Euclidean plane. Each parallelogram is called *unit, tile* or *cell.* each unit has a symbol or value $a_{i,t}$ to represent it's state. The rules specify the relationship of each unit and it's neighbours as follows:

$$a_{i,t}=$$

The variable defines the neighbours of unit $a_{i,t}$; function applied on to determine the value or state of unit $a_{i,t}$. No relationship exists in the lattice system that violates the rules; According to *the mapping list of configurations and symbols*, a symbol could be translated to various configurations, but a configuration corresponds to the unique symbol. There hasn't much limitation about the format of *configuration*. It could be a geometrical shape, an architectural plane , a colored photograph, or a n-dimensional objet which maps to the unit of lattice plane.

The Stephen Wolfram's definition of one dimensional cellular automata could be translated into L grammars. We give the automata which generates the pattern in figure 7 as an example:

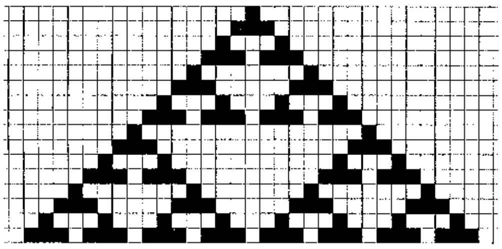


figure 7

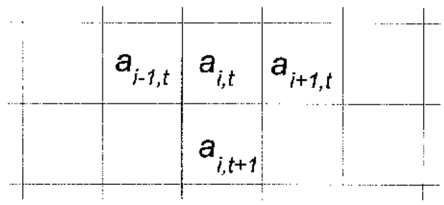


figure 8

The rule ai,t+1=   , where   = [ai-1,t, ai,t, ai+1,t] define the neighbours of ai,t+1.
the mapping list of configurations and symbols.

$a_{i,t+1}$		configuration
0	<---->	
1	<---->	

list 1

The function   is represented as below:

[1,0,0] =1	[1,1,0] =0
[0,0,0] =0	[0,1,0] =1
[1,0,1] =0	[1,1,1] =0
[0,0,1] =1	[0,1,1] =0

Wang tiles could also be translated into L grammars. In figure 9, three tiles is marked as *A,B,C*. For each tile, it's neighbours would be shown in figure 10.

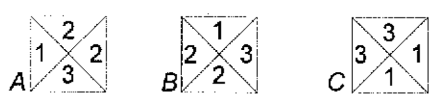


figure 9

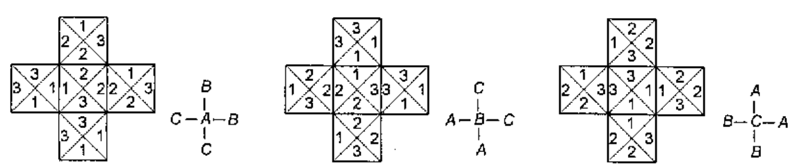


figure 10

It would be translated as follow:

$a_{i,t}=$, $= [\, a_{i-1,t}, a_{i,t-1}, a_{i+1,t}, a_{i,t+1} \,]$

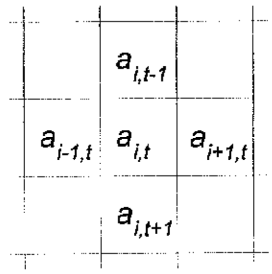


figure 11

The function could be represented as bellows, or illustrated as figure 12.

$[\, C, B, B, C \,] = A$
$[\, A, C, C, A \,] = B$
$[\, B, A, A, B \,] = C$

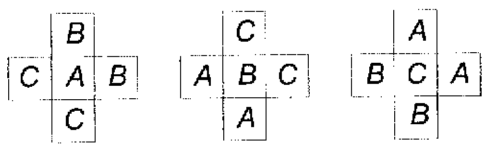


figure 12

In this case, the mapping list of configurations and symbols is not specified.

2-2. n-dimensional L grammars

It wouldn't be difficult to expand the definition of two dimensional L grammars to n-dimensional L grammars. I define n-dimensional L grammars consist of four components:

1.A plane of lattice system.
2.A finite set of symbols.
3.A finite set of rules.
4.A mapping list of configurations and symbols.

The n-dimensional space of lattice system is the n-dimensional space tiled or constructed by the n-dimensional parallelogram periodically, Each n-dimensional parallelogram is called *unit, tile* or *cell* in which each unit has a symbol or value $a_{i,t}$ to represent it's state. The rules specify the relationship of each unit and it's neighbours as this form:

$$a_{i_0, i_1, i_2, \ldots, i_{n-1}} =$$

The variable define the neighbours of unit $a_{i,t}$; function applied on to determine the value or state of unit $a_{i,t}$. According to *the mapping list of configurations and symbols*, a symbol could be translated to various configurations, but a configuration corresponds to the unique symbol.

If the neighborhood is defined in the range of adjacent units, that is , each unit has the only relationships of the units surround it. There wood has a subsystem which include Wang tiles and all the cellular automata mentioned in this paper. In this subsystem, Wang tiles and cellular automata could be translated to each other. Without specified, L grammars always be the subsystem.

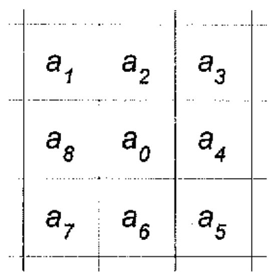


figure 13:In two dimensional condition, a1,a2,a3,a4,a5,a6,a7,a8 are the adjacent units of a0

2-3. Translatability of various lattice system

"All the periodic tiling lattice systems could be translated to a parallelogram system" , and the parallelogram system could be translated to a square system by some transformation such as shifting. T it is obviously that most of the characters of square system is available to other lattice systems.

3. LIMITATIONS OF FINITE STATE GRAMMARS

L grammars, generating configurations by the bottom-up constructive approach, basically are a kind of finite state grammars. some limitation of this approach could not be overcome. e.g. in some structures, one unit might relate to the one far away. This phenomenon is hard to represent in a bottom-up way.

In N. Chomsky's opinion, phrase structure grammars are more powerful than finite state grammars. any sentence generated by the finite state grammars, could also generated by a phrase structure grammar. but sentences generated by phrase structure grammars might not be generated by finite state grammars. Chomsky's work at 1957, had proved that there are three languages couldn't generated by finite state grammars[198] to support the viewpoint above(Chomsky,1957:21-25).

If the problem in 1957 is still unsolvable, L grammars are just a simple-structure grammar but nothing. On the contrary, L grammars could generate Chomsky's three languages if the *sentence* could be seen as a string. Surely the two dimensional finite state grammars are more powerful than the one dimensional one. In the following, we would demonstrate the distinguishability and generation of the second

Chomsky's language—language of mirror image.

3-1. Distinguishing and generating the language

> *language(II):*
> *aa, bb, abba, baab, aaaa, bbbb, aabbaa, abbbba, ...,*
> and in general, all sentences consisting of a string *X* followed by the 'mirror image' of *X* (i.e., *X* in reverse), and only these. (Chomsky,1957:21)

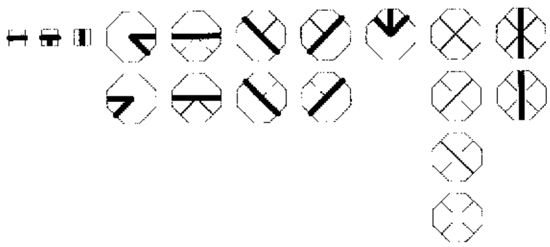


figure 14

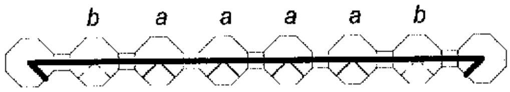


figure 15

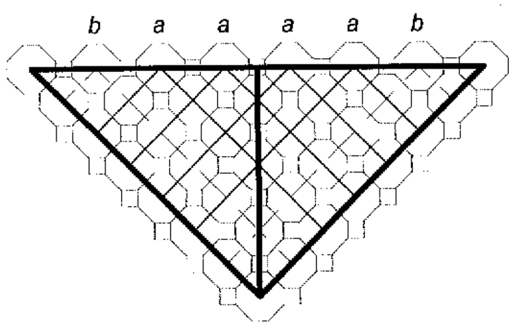
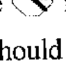


figure 16

Tiles in figure 14 could distinguish or generate language(II). language(II) has only two letters of a and b. We let a and b corresponds to tile and , The tile  is added to the left end of all the string, and  to the right end. The string *baaaab* should correspond to the configuration in figure 15. The result of this tiling would be figure 16.

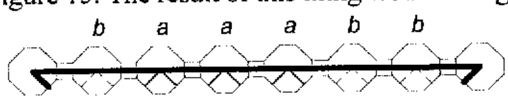


figure 17

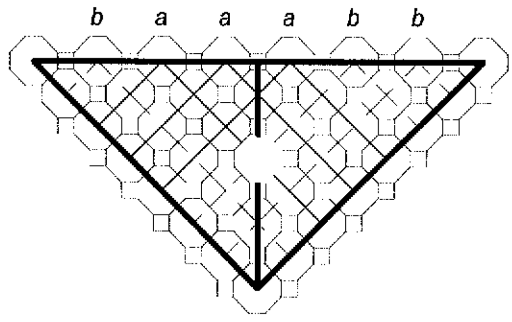


figure 18

The reason why this set of tiles could distinguish and generate language(II) is, each tile *a* would deliver a solid line left-downward and right-downward, and each tile *b* delivers a dash line left-downward and right-downward. The tiles at the same position by mirror would deliver the line meet at the central vertical line. All of the tiles construct the central vertical line are symmetrical. i.e., the solid line meet with the same one at the central line, and so does the dash one. Tiling could be completed only when the left-side string and the right-side string mirror to each other. We give the string *baaabb* for an example, the left-side string *baa* doesn't mirror to the right-side string *abb*, string *baaabb* is not a language(II). string *baaabb* corresponds to the tiles in figure 17. The result of tiling would be the figure 18 which has a absent tile in the center. because there has no proper tile to cover that area to complete the tiling, string *baaabb* is not a language(II).

4.1 GENERATING CONFIGURATIONS

It is clear now that two dimensional L grammars are more powerful than one dimensional one. It would be demonstrated how to generate three dimensional crystal models and two dimensional fractal images of top-down structure.

4.1 Genertating fractal images

The most amazing phenomenon of fractal images is that the complex pattern generated by considerable simple rules. B. B. Mandelbrot (1977:43) specify the *generator* of fractal images in the form in figure 18. Each segment between two spots are replace by the pattern between two larger spots. Finally, the pattern in figure 20 would emerge. The pattern generated by the generator is obviously of a top-down structure. L grammars could generate such a pattern by bottom-up approach, though exhausting.

figure 19 (Mandelbrot, 1977:43)

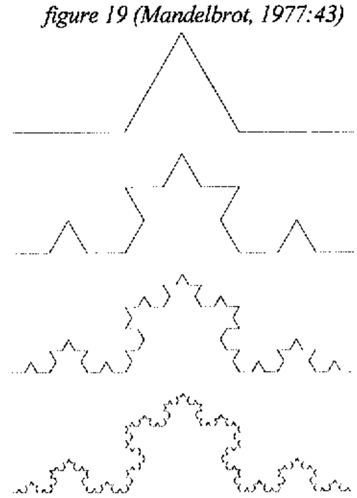
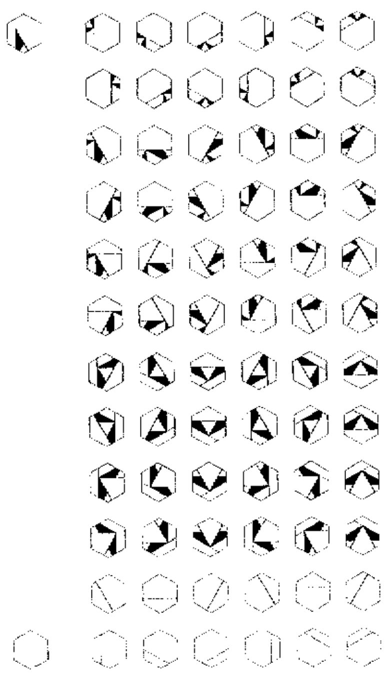


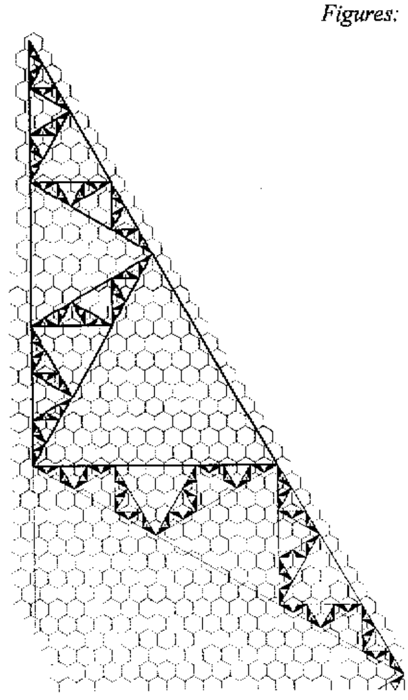
figure 20 (Peitgen & J rgens,1992:91)

4.1.1 The snowflake curve

In figure 22, the 62 hexagonal tiles which, rotated from the nine basic tile in figure 21, could generate the snowflake in figure 23. The pattern and colors, especally the straight lines, must be continuous at the edge of adjacent tiles.

Figures: 21, 22, 23





4.1.2 Sierpinski gasket

The tiles in figure 24 could also generate the Sierpinski gasket pattern in figure 25.

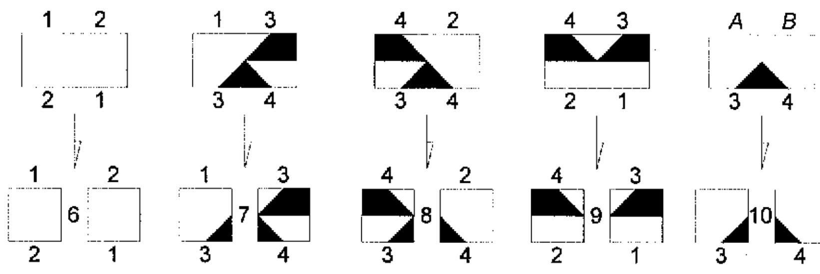


figure 24

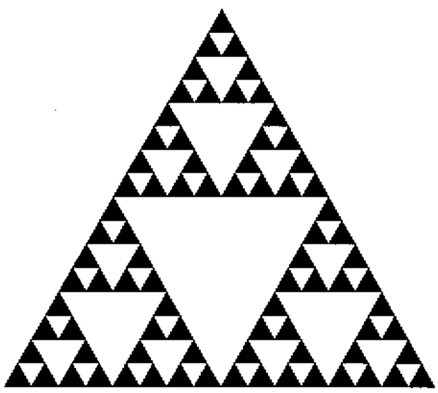


figure 25

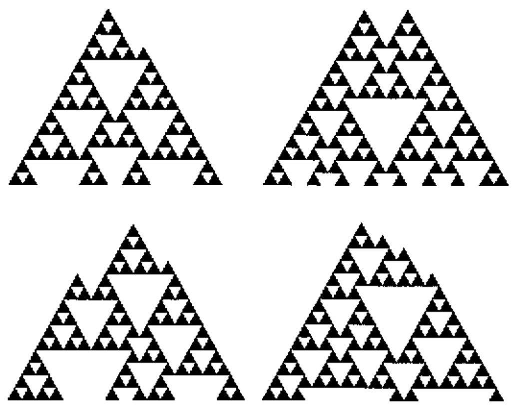


figure 26

If we let *A* to be 1 and *B* to be 2, then the tiling of these tiles would be undetermined. Excep the typical pattern in figure 25, there would have lots of similuar patterns in figure 26

4.2 Generating three dimensional configurations

The defination of two dimensional cellular automata and tiles are easy to expend to genera three or higher dimensional configurations. For example, the crystal model imagined by French mineralogist Ren Just Ha y could generat by a grammar whic consider the summation of itself and all the adjacent units(figure 27). If the summation is 0, the value of central unit would be 0. Otherwise, the value of central unit would be 1.

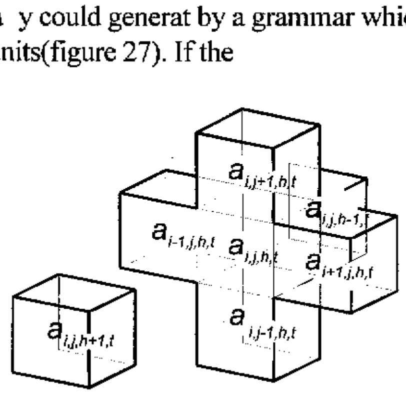


figure 27

$$a_{i,j,h,t+1} = \left[a_{i-1,j,h,t}+a_{i,j-1,h,t}+a_{i,j,h-1,t}+a_{i,j,h,t}+a_{i+1,j,h,t}+a_{i,j+1,h,t}+a_{i,j,h+1,t} \right]$$

7	6	5	4	3	2	1	0
↓	↓	↓	↓	↓	↓	↓	↓
1	1	1	1	1	1	1	0

List 2

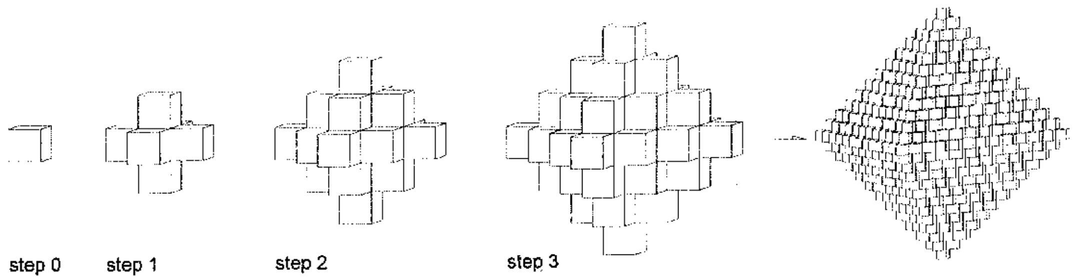


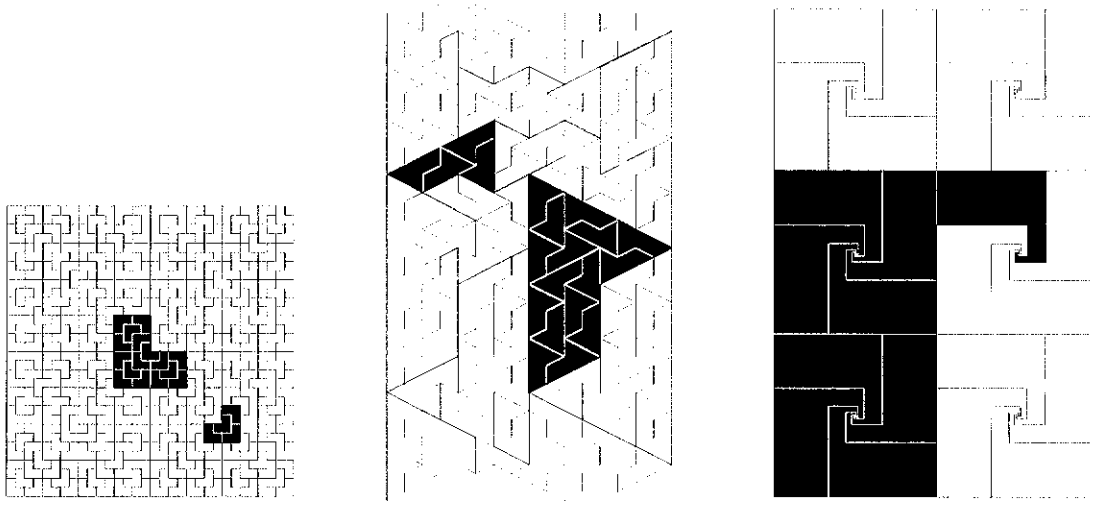
figure 28 a growing up crystal whose initial state is a unit be valued 1.

5.0 LIMITATIONS OF L GRAMMARS

5.1 Can't generate arbitrary scaled configuration

Some tiles could generate different scaled shapes such as parallelogram, triangle and some specificic shapes(figure 29(a),(b),(c),(d),(e))

(a) (b)



(c) (d) (e)

figure 29 (c),(d),(e)(Gr nbaum & Shephard,1987:523)

Cases in figure 29 could not generate different scaled shapes without others. In fact, shapes in (c) and (d) are generated by top-down approach instead of bottom-up one (see figure 30). i.e., the larger shapes are not constructed by the smaller one. On the contrary, smaller shapes devided from the larger one. Mostly, tiles can't exactly construct scaled shapes.

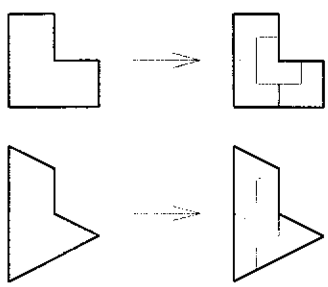


figure 30

If there have more specific restrictions, tiles could generate exactly some scaled shapes. The set of tiles in figure 30 could generate *the segmental Fibonacci golden spiral*.

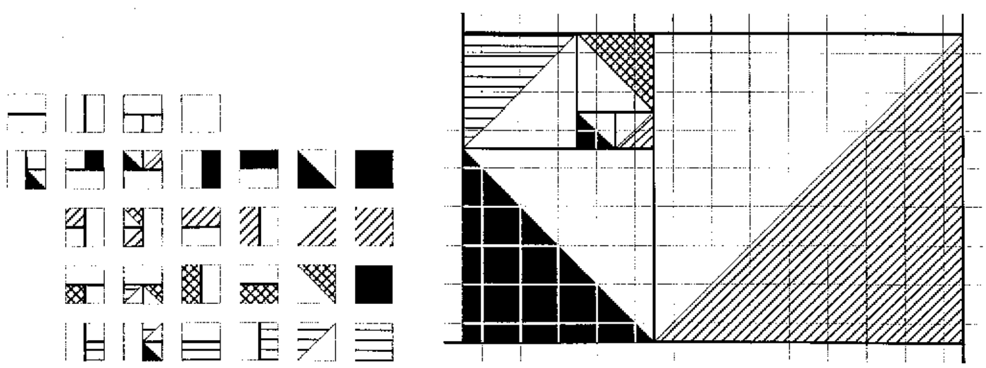


figure 31

But the tiles still can't generate golden spiral. because the golden spiral is consisted of various scaled quarter circle. It would be found that we need infinite tiles of different pattern to generate infinite golden spiral. And this would violate the definition.

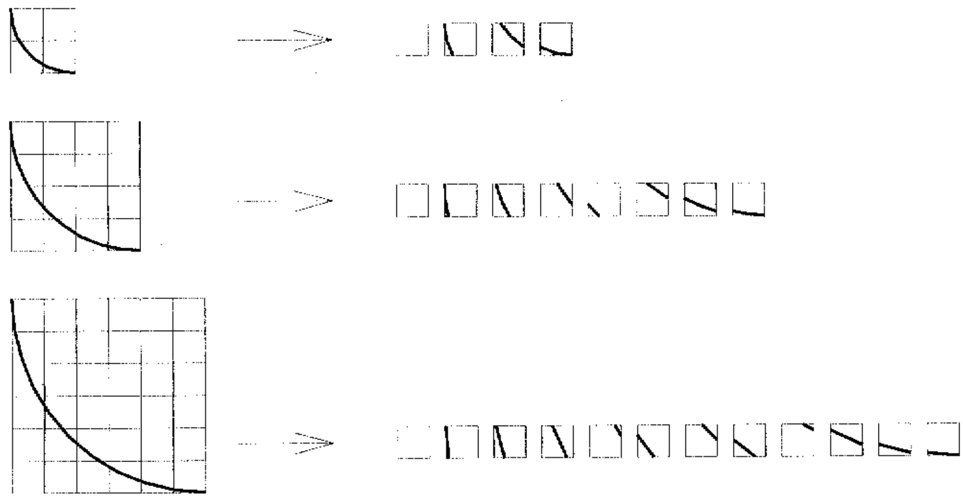


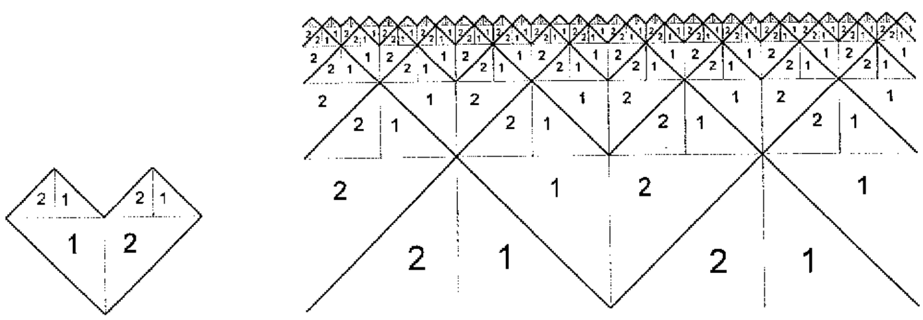
figure 32／different sacled curve genertated by different set of tiles.

Once a shape has a subshape of curve or circle, It would be very difficult to generate that shape in arbitrary scale. Besides, fractal images are also the difficult one as being self-similar. Considering these difficulties, some alternatives would be proposed in the following.

6.0 ALTERNATIVE WAYS TO GENERATE CONFIGURATIONS

6.1 Inflating/deflating tiles

Inflating/deflating tiles solves the arbitrary scaled problem directly by a unconstent lattice system which is inflating or deflating (figure 33).



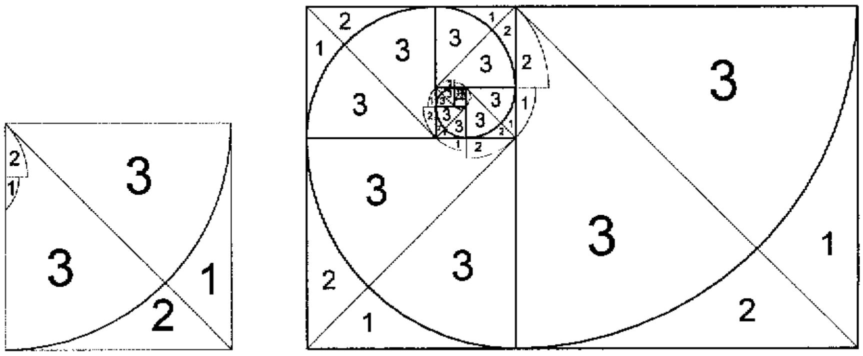
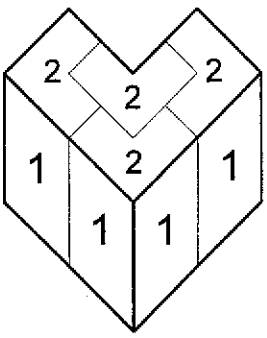


figure 33

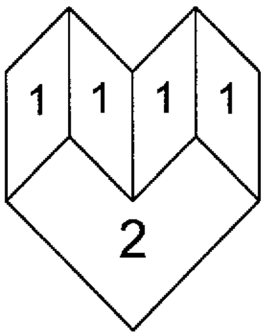
6.2 Top-down—replaceable tiles

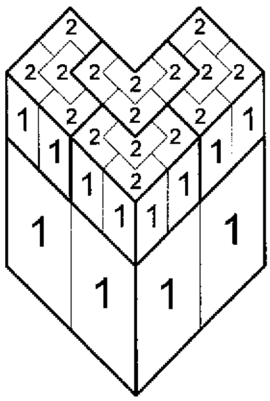
In figure 30, there are two typical top-down ways—a set of smaller tiles replace the larger one. They would be similar to the inflating/deflating tiles, If we take a viewpoint of figure 34 and 35.



top-view

figure 34

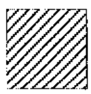
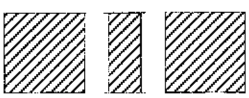
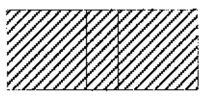




bottom-view

figure 35

6.3 Flexible L grammars

If the lattice plane is flexible, the boundry of a unit would depend on the configuration it correspond to. For example, a_0 corresponds to , a_2 to , and a_4 to , then the configuration of the union of a_0,a_2,and a_4 , instead of  , would be . one condition, the height of a_1,a_3 and a_2 should be consistent.

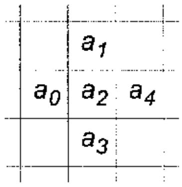


figure 36

6.4 Turing plotter

Turing plotter consist of a normal turing machine and a plotter. First, Turing machine executes a calculation and deliver the output to the plotter. According to the received data, plotter would plot or erase on the plane. When the Turing plotter stoped, the generation is completed, and the pattern on the plane would be the output.

Turing plotter could generate various configurations. It also could tile the plane as the plotter is exchanged to a tiler. In figure 37 the tiler tiles the infinite plane spirally with three tiles .

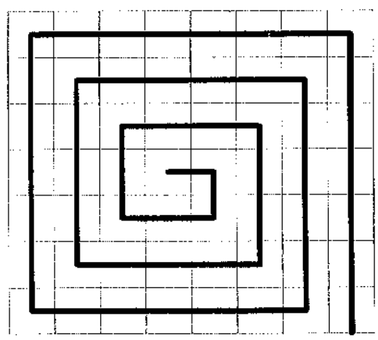


figure 37

BIBLIOGRAPHY

Chomsky, Noam 1957 *Syntactic Structures* Paris Mouton publishers

Gips, James Elliot 1974 *Shape Grammars and Their Uses*, Stanford University, Ph.D, Computer Science

Grunbaum, Branko & Shephard, G. C. 1986 *Tilings and Patterns* New York :W. H. Freeman and company

Mandelbrot, Benoit B. 1977 *The Fractal Geometry of Nature* New York : W. H. Freeman and company

March,Lionel & Steadman, Philip 1971 *The Geometry of Environment* London ; RIBA

Mitchell, W. 1990 "Languages of Architectural Form" *The Logic of Architecture: Design, Computation, and Cognition*, Cambridge: MIT, pp.131-148

Peitgen, Heinz-Otto & J rgens, Hartmut & Saupe, Dietmar 1992 *Chaos and Fractals-New Frontiers of Science*, New York : Springer-Verlag

Wang, Hao 1965 "Games, Logic and Computers", *Scientific American.* November: pp 98-106

Wolfram, Stephen 1984 a'Computer Software in Science and Mathematics', *Scientific American.* September: pp 140-151

---

195 Not only generating the shape, each rule and stage of shape grammars but also applid at architecture ought to correspond to the architectural meaning of that architecture. Because this analysis is to understand or interpret the meaning of architectural form, terseness and facility would be necessary.

196 In formal language, the term machine is identity to grammar. I call L grammars as a grammar, though L grammars is a finite state automata and hasn't the apparent linguistic structure. The finite state automata is also called finite state grammar by N. Chomsky.

197 rotation, reflection, and scale are forbidden, except translation.

198 The finite state grammars mentioned by Chomsky ought to be the one dimensional one. And the three languages couldn't generated by finite state grammars is :

(I)*ab, aabb, aaabbb*, ..., and in general, all sentences consisting of *n* occurrences of *a* followed by *n* occurrences of *b* and only these;

(II)*aa, bb, abba, baab, aaaa, bbbb, aabbaa, abbbba*, ..., and in general, all sentences consisting of a string *X* followed by the 'mirror image' of *X* (i.e., *X* in reverse), and only these;

(I)*aa, bb, abab, baba, aaaa, bbbb, aabaab, abbabb*, ..., and in general, all sentences consisting of *a* string *X* of *a*'s and *b*'s followed by the identical string *X*, and only these.

Chomsky has proved that phrase structure grammars could generate the language(I) and (II), but fail to language(III). (Chomsky,1957)