From Aesthetically Appealing Chinese Lattices To Routing or Circuitry Systems using Shape Grammar-Cellular Automata Methodology

NKS 2007 Conference
Wolfram Science

Thomas H. Speller, Jr.
Doctoral Candidate
Engineering Systems Division
School of Engineering
MIT

July 14, 2007

Today's Topic

- Introduction to system architecture
  - Motivation
- Introduction to shape grammar and the SG→CA methodology
- Chinese lattices generated by SG→CA
- Routing problem: underfloor heating system using SG→CA

Current Discipline of System Architecture is in part Limited by

- Too few concept alternatives considered
  - Limited time and budget
- Dominance of paradigms, subjective personalities, political positions and financial influencers (*The Structure of Scientific Revolutions,* Thomas Kuhn, 1970*)*
  - Individuals
  - Teams
  - Enterprises
- Insufficient interaction of concept design and selection with stakeholders to elicit their true wants
- Compulsion to do rather than think, create alternatives, evaluate and rank alternatives, iterate system architectures with stakeholders

Recommend reading Skunk Works by Ben Rich

Motivation

- **System architecture**

1. To generate a creative space of system architectures that are physically legitimate and satisfy a given specification inspired by nature's bottom-up self-generative processes
   – using a shape grammar and cellular automata approach
2. Expanding the application of the SG→CA approach to the domains -- study of Chinese lattice-meanders and of underfloor heating systems design
   - modeling complex, nonlinear physical phenomena;
     Science → Application

Shape Grammar

- Based on transformational grammars [N. Chomsky 1957] , which generate a language of one dimensional strings
- Shape grammars (Stiny, 1972; Knight, 1994, Stiny 2006)
  - are systems of rules for characterizing the composition of designs in spatial languages (nondeterminant)
  - The grammar is unrestricted having the capability of producing languages that are recursively enumerable
  - defined by a quadruple SG = {V_T, V_M, R, I)}, generate a language of two or even three dimensional objects that are composed of an assemblage of terminal shapes, where
    - V_T is a set of terminal shapes (i.e., terminal symbols)
    - V_M is a set of markers (i.e., variables)
    - R is a set of shape rules (addition/subtraction and Euclidean transformations), u→v is the shape rule (i.e., productions; a production set of rules specifies the sequence of shape rules used to transform an initial shape to a final state and thus constitutes the heart of the grammar)
      - u is in $(V_M \cup V_T)^+$ and v is in $(V_M \cup V_T)^*$
    - I is the initial shape to which the first rule is applied (i.e., start variable)

The System Architecture Generative Algorithm in Stages: SG→CA

Given: Specification of function, constraints, requirements and "ilities" in solution neutral form

Stages *1:* Developing the shape grammar design system to describe the system architecture

2: Adapting the shape grammar to a cellular automaton neighborhood

3: Developing the computational system (CA rules, combinatorics, algorithms) to generate the system architecture (from primitives through lower- and higher-order modules)

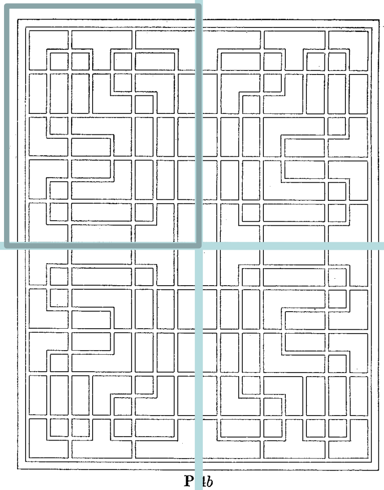
4: (Optional) Narrowing the creative or solution space of system architectures by symmetry grouping or fitness/survival tests

5: (Optional) Configuring the complete system architecture

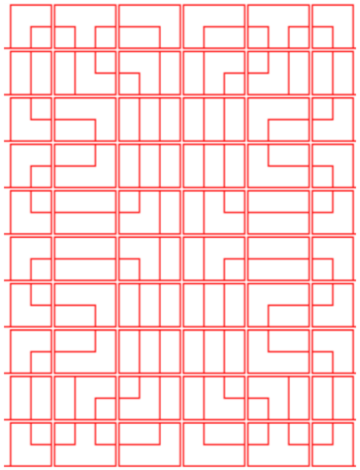
6: Selecting from the creative or solution space of system architectures by the stakeholders

Analyzing the Style of Chinese Lattices and Meanders using Shape Grammar

Chinese lattice[1]



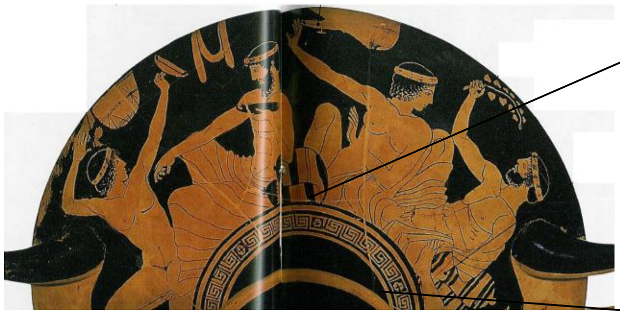
SG→CA generated Chinese lattice



Look for shapes:
- repeating patterns
- order, uniformity (interrelationships, modularity),
- symmetries and asymmetries,
- primitive shapes

1 Dye, D.S., *A Grammar of Chinese Lattice. Vol. II. 1937, Cambridge, Mass.: Harvard-Yenching Institute, p. 235*

ancient Greek meanders



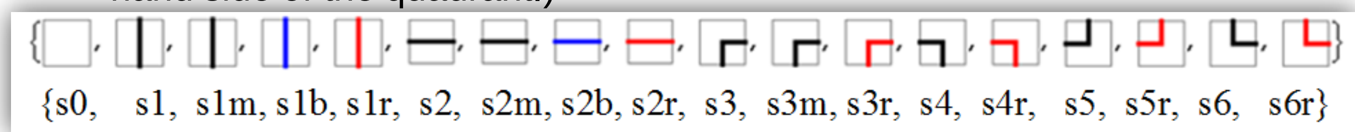
Shape Grammar-Cellular Automata Methodology[1] (SG→CA) for Meanders

- The given specification
  - Start in the upper right hand corner and traverse every cell once without crossing lines. Exit the lower left hand corner. Discontinuities are allowed.
- Shape Grammar
  - Shape variables:= {Primitives, Modules, Markers}
  - Initial condition, configuration
- The generating machine is a two-dimensional cellular automaton
  - Transcribe the shapes into symbols, then
  - Compute generatively the system architectures by computing with the symbolically represented shapes
- Translate back to shape and provide graphical visual output

[1]Described in Speller, T.H., Jr., D. Whitney, and E. Crawley, *Using Shape Grammar to Derive Cellular Automata Rule Patterns. Complex Systems, 2007.* ***17: p. 79-102.***

Stages 1 and 2: Developing a Shape Grammar and Neighborhood for a Meander Specification

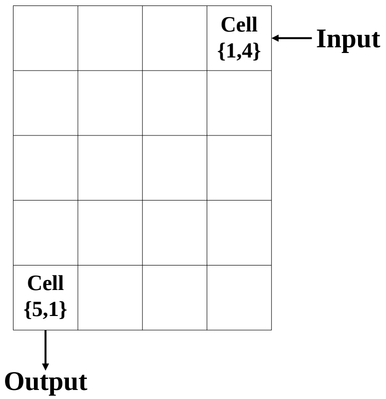
- Shape variables:
  - 1 empty shape, symbolically: {s0}
  - 6 shape variables (black shapes), symbolically: {s1, s2, s3, s4, s5, s6}
  - 11 shape markers: (boundary and colored shapes),
    - symbolically: {s1m, s1b, s1r, s2m, s2b, s2r, s3m, s3r, s4r, s5r, s6r}.
    - (The __m designates a marker that constrains the top row and left-hand boundaries. The __b represents a blue marker to bound the upper right or lower left corners of the quadrant. The __m and __b markers are used in the initial conditions. The __r or red markers are rule generated, serving as boundaries for the bottom row and the right-hand side of the quadrant.)



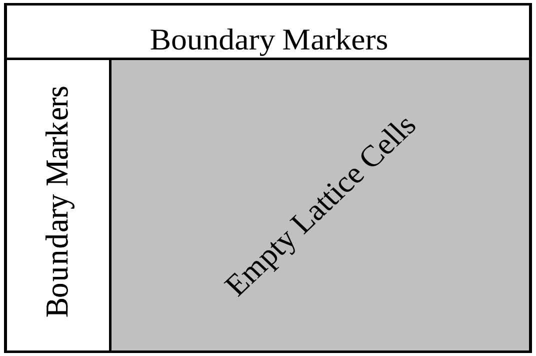
- 128 rules: the formal simple relationships of form-function symbolically expressed according to local neighborhood conditions, classified as Group I rules when the conditional is invariant or as Group II rules when options exist for the condition

Generating a Catalog Based on a System Architecture Style: Lattice-meander

The Specification Indicating the Input and Output Cell Positions

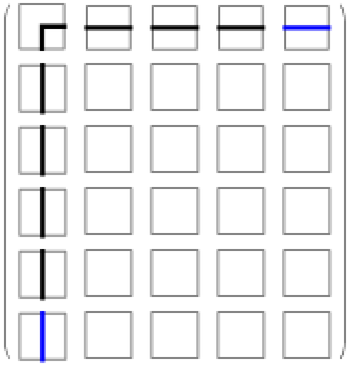


Lattice at Initial Condition

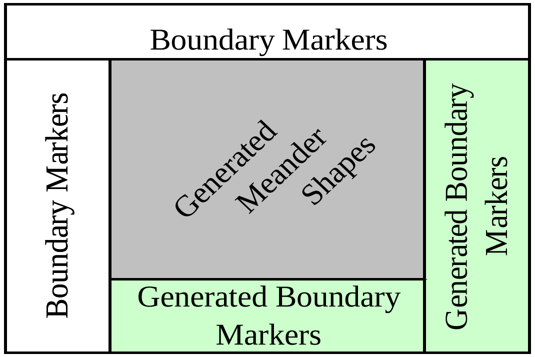


The Initial Condition in Graphical and Symbolic Form

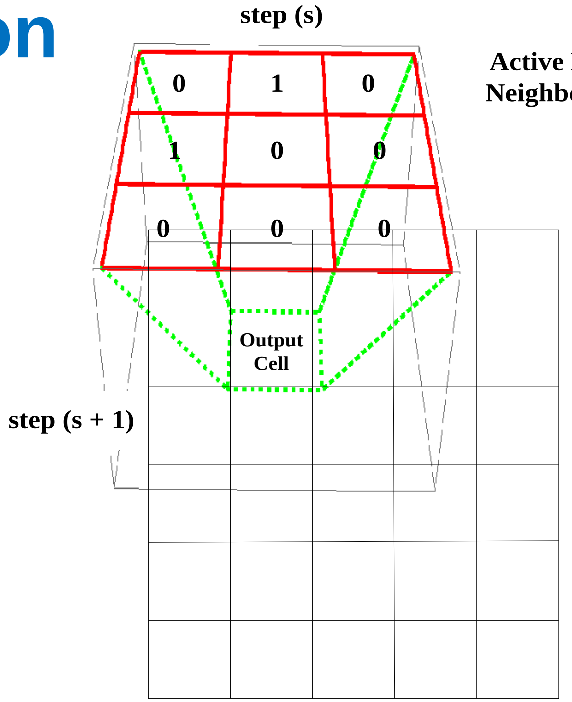
$$\begin{pmatrix} s3m & s2m & s2m & s2m & s2b \\ s1m & s0 & s0 & s0 & s0 \\ s1m & s0 & s0 & s0 & s0 \\ s1m & s0 & s0 & s0 & s0 \\ s1m & s0 & s0 & s0 & s0 \\ s1b & s0 & s0 & s0 & s0 \end{pmatrix} \equiv$$



Lattice after Meander Generation

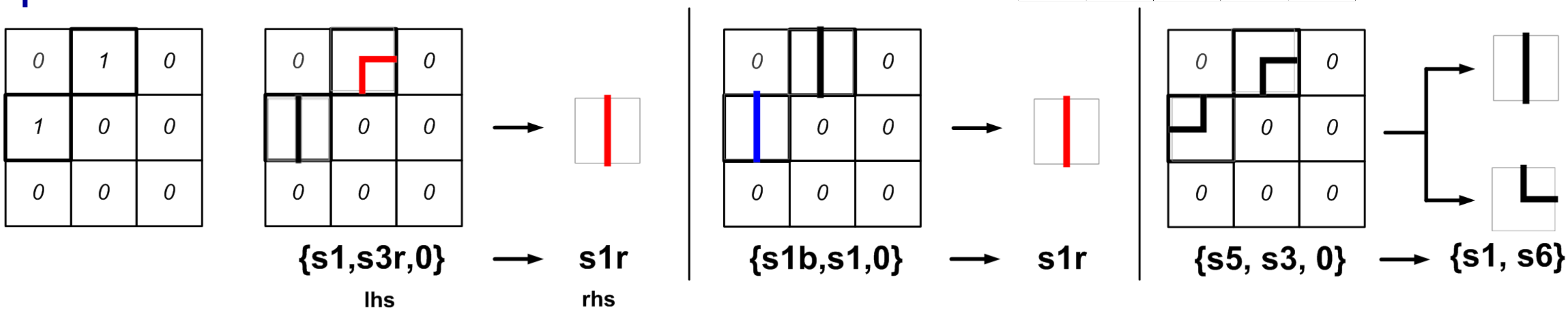


Stage 3 – Developing the Cellular Automaton, Combinatorics and Algorithm for the Meander Specification



- The Active Moore 2-Dimensional Neighborhood (only 2 cells are active)

- 3 rule examples



Enumerating the Rule Sets

- Rule set:= {Group I, Group II}
  - Group I is invariant, size 106 with single output
  - Group II is size 22 having two outputs
- 2^{22} rule set combinations (4,194,304)

```
GroupI = {{s3m} → s3m, {s1m} → s1m, {s2m} → s2m, {s1b} → s1b, {s2b} → s2b, {s0, s0, 0} → s0, {s0, s2b, 0} → s0,
    {s0, s2m, 0} → s0, {s1, s1r, 0} → s1r, {s1, s2, 0} → s3, {s1, s2r, 0} → halt, {s1, s3r, 0} → s1r, {s1, s4r, 0} → s1r,
    {s1, s5, 0} → s3, {s1, s5r, 0} → halt, {s1, s6, 0} → s3, {s1b, s0, 0} → s0, {s1b, s6, 0} → s3r, {s1m, s0, 0} → s0,
    {s1m, s2m, 0} → s3, {s1m, s6, 0} → s3, {s1r, s1, 0} → s6r, {s1r, s2, 0} → halt, {s1r, s3, 0} → s6r, {s1r, s4, 0} → s6r,
    {s1r, s5, 0} → halt, {s1r, s6, 0} → halt, {s2, s1, 0} → s5, {s2, s1r, 0} → s5r, {s2, s2b, 0} → s2r, {s2, s2r, 0} → s4r,
    {s2, s3, 0} → s5, {s2, s3r, 0} → s5r, {s2, s4, 0} → s5, {s2, s4r, 0} → s5r, {s2, s5r, 0} → s4r, {s2r, s1, 0} → s5r,
    {s2r, s1r, 0} → s5r, {s2r, s2, 0} → s2r, {s2r, s3, 0} → s5r, {s2r, s4, 0} → s5r, {s2r, s4r, 0} → s5r, {s2r, s5, 0} → s2r,
    {s2r, s5r, 0} → halt, {s2r, s6, 0} → s2r, {s3, s1, 0} → s5, {s3, s1r, 0} → s5r, {s3, s2b, 0} → s2r, {s3, s2r, 0} → s4r,
    {s3, s3, 0} → s5, {s3, s3r, 0} → s5r, {s3, s4, 0} → s5, {s3, s4r, 0} → s5r, {s3, s5r, 0} → s4r, {s3r, s1, 0} → s5r,
    {s3r, s2, 0} → s2r, {s3r, s3, 0} → s5r, {s3r, s4, 0} → s5r, {s3r, s5, 0} → s2r, {s3r, s6, 0} → s2r, {s4, s1r, 0} → s1r,
    {s4, s2, 0} → s3, {s4, s2b, 0} → s3r, {s4, s2m, 0} → s3, {s4, s2r, 0} → halt, {s4, s3r, 0} → s1r, {s4, s4r, 0} → s1r,
    {s4, s5, 0} → s3, {s4, s5r, 0} → halt, {s4, s6, 0} → s3, {s5, s1r, 0} → s1r, {s5, s2, 0} → s3, {s5, s2r, 0} → halt,
    {s5, s3r, 0} → s1r, {s5, s4r, 0} → s1r, {s5, s5, 0} → s3, {s5, s5r, 0} → halt, {s5, s6, 0} → s3, {s5r, s1, 0} → s6r,
    {s5r, s1r, 0} → halt, {s5r, s2, 0} → halt, {s5r, s3, 0} → s6r, {s5r, s4, 0} → s6r, {s5r, s4r, 0} → halt, {s5r, s5, 0} → halt,
    {s5r, s5r, 0} → halt, {s5r, s6, 0} → halt, {s6, s1, 0} → s5, {s6, s1r, 0} → s5r, {s6, s2r, 0} → s4r, {s6, s3, 0} → s5,
    {s6, s3r, 0} → s5r, {s6, s4, 0} → halt, {s6, s4r, 0} → halt, {s6, s5r, 0} → s4r, {s6r, s1, 0} → s5r, {s6r, s1r, 0} → s5r,
    {s6r, s2, 0} → s2r, {s6r, s3, 0} → s5r, {s6r, s4, 0} → halt, {s6r, s4r, 0} → halt, {s6r, s5, 0} → s2r, {s6r, s5r, 0} → halt,
    {s6r, s6, 0} → s2r, {s1b, s1, 0} → s1r, {s1b, s3, 0} → s1r};

GroupII = {{s1, s1, 0} → {s1, s6}, {s1, s3, 0} → {s1, s6}, {s1, s4, 0} → {s1, s6}, {s2, s2, 0} → {s2, s4}, {s2, s5, 0} → {s2, s4},
    {s2, s6, 0} → {s2, s4}, {s3, s2, 0} → {s2, s4}, {s3, s5, 0} → {s2, s4}, {s3, s6, 0} → {s2, s4}, {s4, s1, 0} → {s1, s6},
    {s4, s3, 0} → {s1, s6}, {s4, s4, 0} → {s1, s6}, {s5, s1, 0} → {s1, s6}, {s5, s3, 0} → {s1, s6}, {s5, s4, 0} → {s1, s6},
    {s6, s2, 0} → {s2, s4}, {s6, s5, 0} → {s2, s4}, {s6, s6, 0} → {s2, s4}, {s1m, s3, 0} → {s6, s1}, {s1m, s1, 0} → {s6, s1},
    {s3, s2m, 0} → {s4, s2}, {s2, s2m, 0} → {s4, s2}};
```

Cellular Automaton Generates Each Meander

Example:

Step 0

s3m	s2m	s2m	s2m	s2b
s1m	s0	s0	s0	s0
s1m	s0	s0	s0	s0
s1m	s0	s0	s0	s0
s1m	s0	s0	s0	s0
s1b	s0	s0	s0	s0

Step 1

s3m	s2m	s2m	s2m	s2b
s1m	s3	s0	s0	s0
s1m	s0	s0	s0	s0
s1m	s0	s0	s0	s0
s1m	s0	s0	s0	s0
s1b	s0	s0	s0	s0

Sequence of CA generation Step 2

s3m	s2m	s2m	s2m	s2b
s1m	s3	s2	s0	s0
s1m	s6	s0	s0	s0
s1m	s0	s0	s0	s0
s1m	s0	s0	s0	s0
s1b	s0	s0	s0	s0

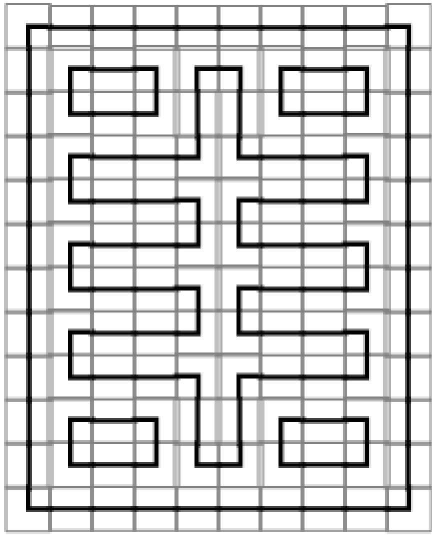
Step 6 Completes the upper left quadrant

s3m	s2m	s2m	s2m	s2b
s1m	s3	s2	s4	s3r
s1m	s6	s2	s5	s1r
s1m	s3	s2	s2	s5r
s1m	s6	s2	s2	s4r
s1b	s3r	s2r	s2r	s5r

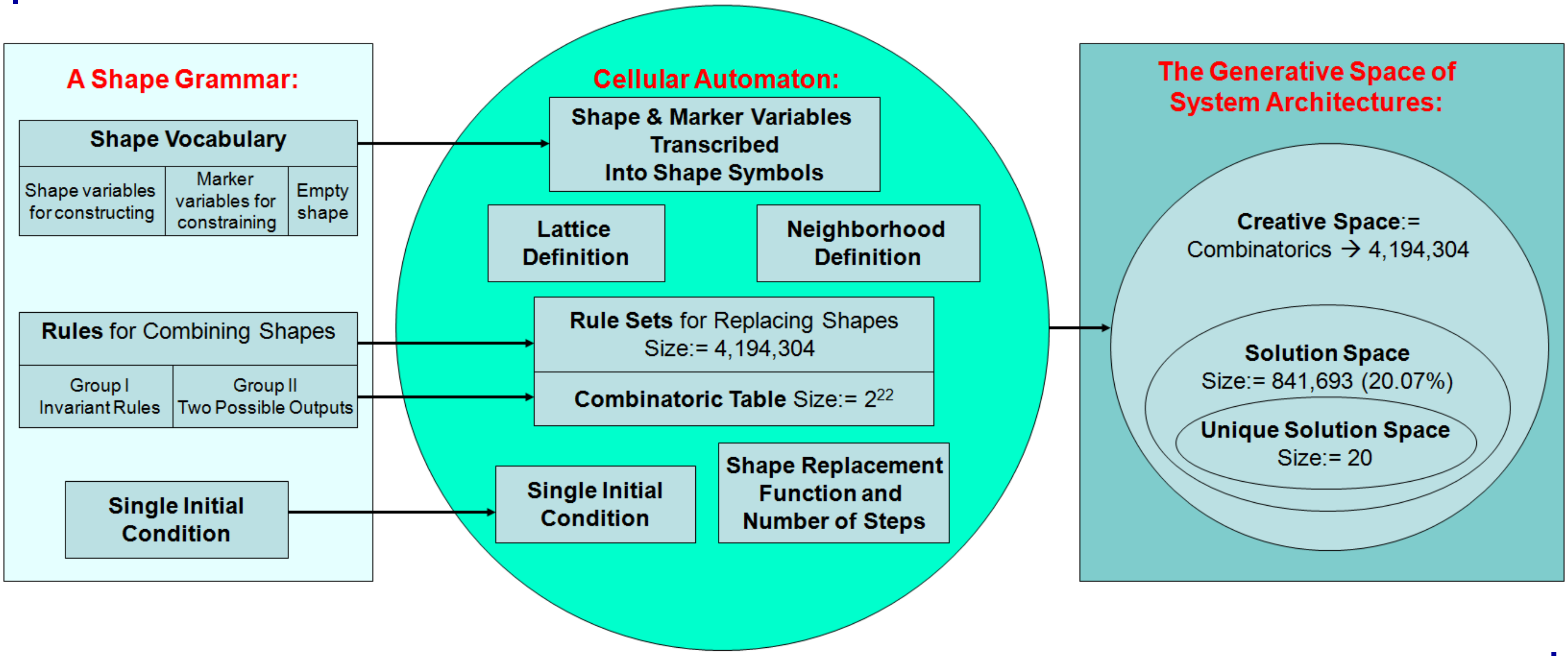
Applying symmetry rules generates upper half and bottom half of meander.
Result for this example:

s3m	s2m	s2m	s2m	s2b	s2	s2	s2	s2	s4
s1m	s3	s2	s4	s3r	s4	s3	s2	s4	s1
s1m	s6	s2	s5	s1r	s1	s6	s2	s5	s1
s1m	s3	s2	s2	s5r	s6	s2	s2	s4	s1
s1m	s6	s2	s2	s4r	s3	s2	s2	s5	s1
s1b	s3r	s2r	s2r	s5r	s6	s2	s2	s4	s1
s1	s6	s2	s2	s4	s3	s2	s2	s5	s1
s1	s3	s2	s2	s5	s6	s2	s2	s4	s1
s1	s6	s2	s2	s4	s3	s2	s2	s5	s1
s1	s3	s2	s4	s1	s1	s3	s2	s4	s1
s1	s6	s2	s5	s6	s5	s6	s2	s5	s1
s6	s2	s2	s2	s2	s2	s2	s2	s2	s5

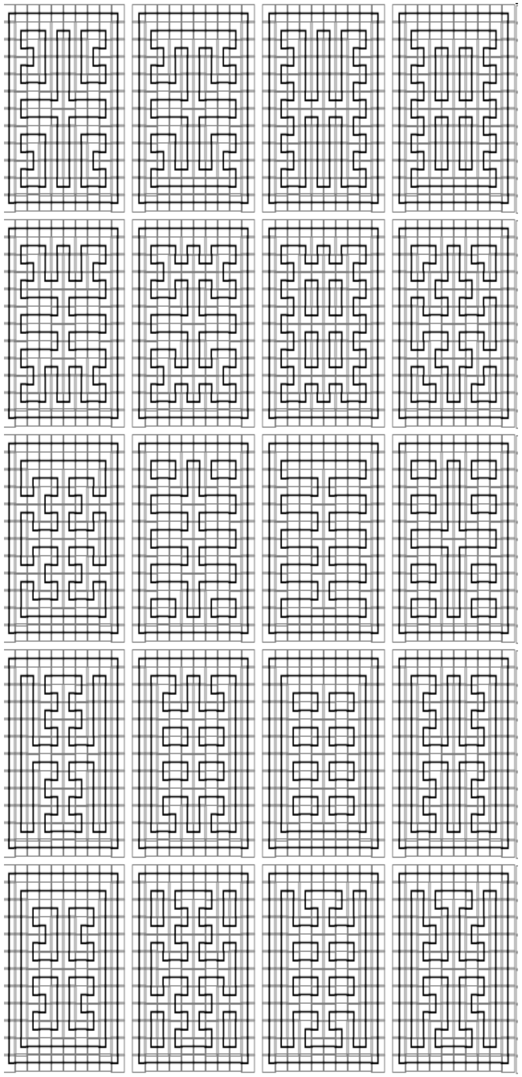
≡



Generating a System Architecture for a Circuit or Routing Specification

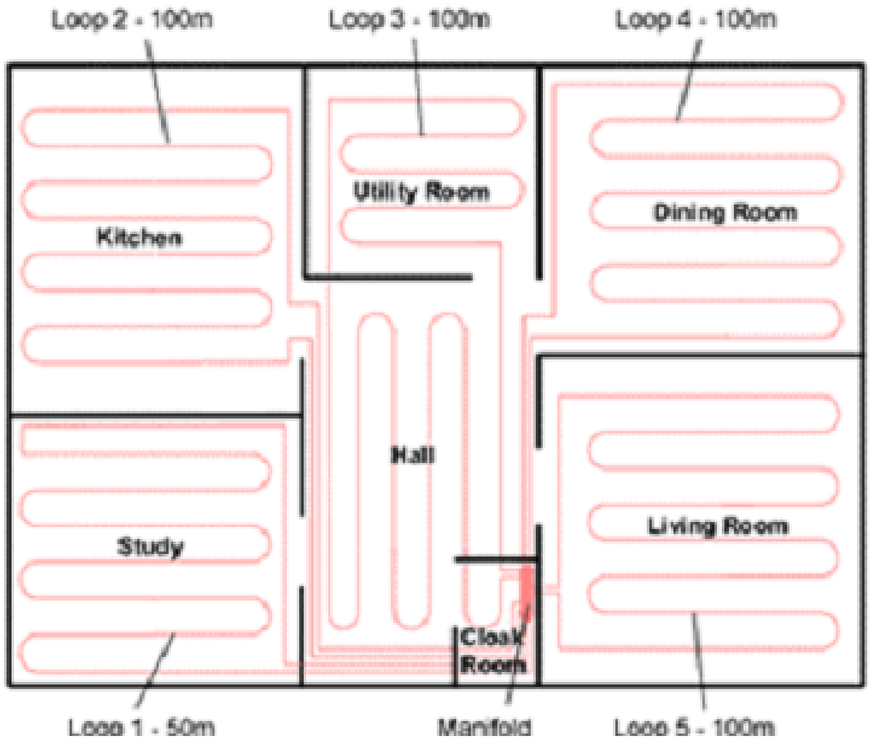


The 20 Unique Solutions (Architectures) Satisfying the Specification



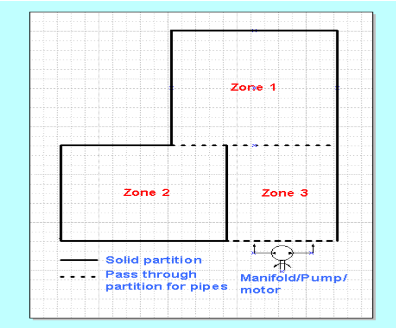
An Application of the Meander Style: A Routing or Circuitry Problem

Example of a Typical Underfloor Heating System Pipe Layout



www.underfloorheatingsystems.co.uk

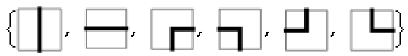
Specification– Configuring the Complete Underfloor Heating Supersystem Containing 3 Modules (zones)



1. There are three zones as shown in Figure →
2. The manifold/pump may be located anywhere at the bottom of zone 3
3. From zone 3 the input/output piping may connect anywhere to zone 1, but may not connect to zone 2 because there is a solid foundation wall between zone 2 and zone 3
4. Zones 1 and Zone 2 interconnect
5. The system architecture(s) generated must be efficient for providing uniform heat and be the lowest cost to manufacture and install

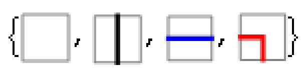
Stages 1 and 2 – Shape Grammar and Neighborhood Development for an Underfloor Heating System

- Shape variables:
  - 1 empty shape, symbolically: {s0}
  - 6 shape variables symbolically: {s1, s2, s3, s4, s5, s6}
  - 37 shape markers:



{s1, s2, s3, s4, s5, s6}

    - symbolically: {s0m, s0r, s1m, s1mr, s1r, s1v, s1vr, s1p, s1n, s2m, s2mr,
    - s2r, s2b, s2br, s2p, s2n, s3m, s3r, s3p, s3n, s4r, s4v, s4vr, s4p, s4n, s5m,
    - s5mr, s5r, s5v, s5vr, s5p, s5n, s6m, s6mr, s6r, s6p, s6n}
    - (The additional letters used to designate some of the shape markers are required because of greater boundary conditions stemming from the use of three zones or matrices instead of just one.) e.g.,



{s0m, s1m, s2b, s4r}

- 432 rules: the formal simple relationships of form-function symbolically expressed according to local neighborhood conditions

Stage 3 – Developing the Cellular Automata, Combinatorics and Algorithm

The Local Neighborhood Definition

0	*1*	*0*
1	*1*	*0*
0	*0*	*0*

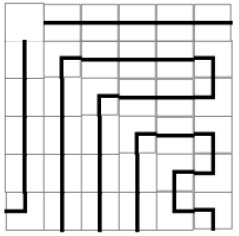
Initial Condition Options for Zone 1
(2^{19}, 524,288 Possible Initial Conditions, 34 solutions)

s0m	s2m	s2m	s2m	s2m	s2b
s1m	s0	{s0, s0r}	{s0, s0r}	{s0, s0r}	s0
s1m	{s0, s0r}	{s0, s0r}	{s0, s0r}	{s0, s0r}	s0
s1m	{s0, s0r}	{s0, s0r}	{s0, s0r}	{s0, s0r}	s0
s1m	{s0, s0r}	{s0, s0r}	{s0, s0r}	{s0, s0r}	s0
s5m	{s0, s0r}	{s0, s0r}	{s0, s0r}	{s0, s0r}	s0

e.g.,
generated
In 9 CA steps

s0m	s2m	s2m	s2m	s2m	s2b
s1m	s3	s2	s2	s2	s4r
s1m	s1	s3	s2	s2	s5r
s1m	s1	s1	s3	s2	s4r
s1m	s1	s1	s1	s3	s5r
s5m	s1n	s1p	s1	s6	s4r

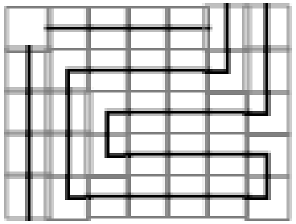
≡



Initial Condition Options for Zone 2
(2^{19}, 524,288 Possible Initial Conditions, 37 solutions)

s0m	s2m	s2m	s2m	s2m	s1n	s1p
s1m	s0	{s0, s0r}	{s0, s0r}	{s0, s0r}	{s0, s0r}	s0
s1m	{s0, s0r}	{s0, s0r}	{s0, s0r}	{s0, s0r}	{s0, s0r}	s0
s1m	{s0, s0r}	{s0, s0r}	{s0, s0r}	{s0, s0r}	{s0, s0r}	s0
s1m	{s0, s0r}	{s0, s0r}	{s0, s0r}	{s0, s0r}	{s0, s0r}	s0

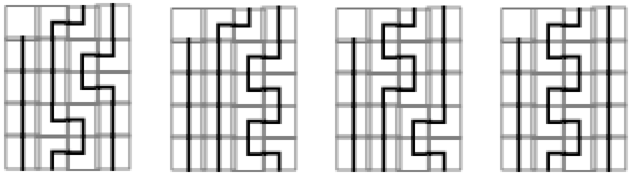
e.g., generated
In 9 CA steps



Initial Condition Options for Zone 3
(2^{8}, 256 Possible Initial Conditions, 4 solutions)

s0m	s3	s5	s1r
s1m	{s0, s0r}	{s0, s0r}	s0
s1m	{s0, s0r}	{s0, s0r}	s0
s1m	{s0, s0r}	{s0, s0r}	s0
s1m	{s0, s0r}	{s0, s0r}	s0

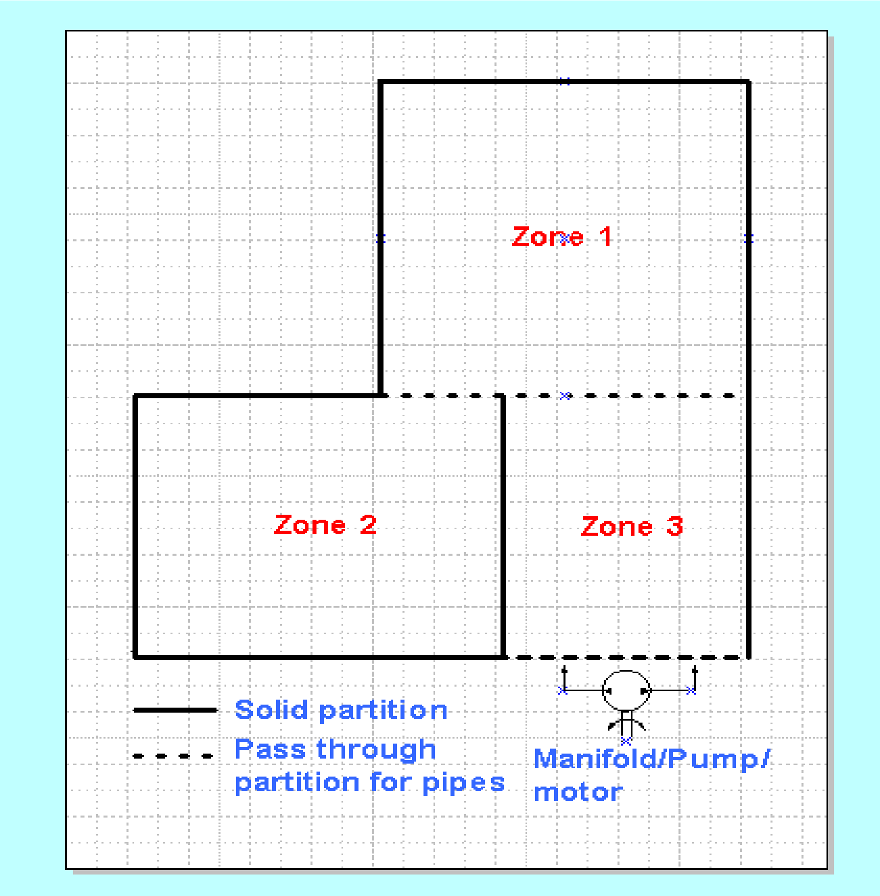
e.g., generated
In 6 CA steps



Stage 4 – Narrowing the Creative Space

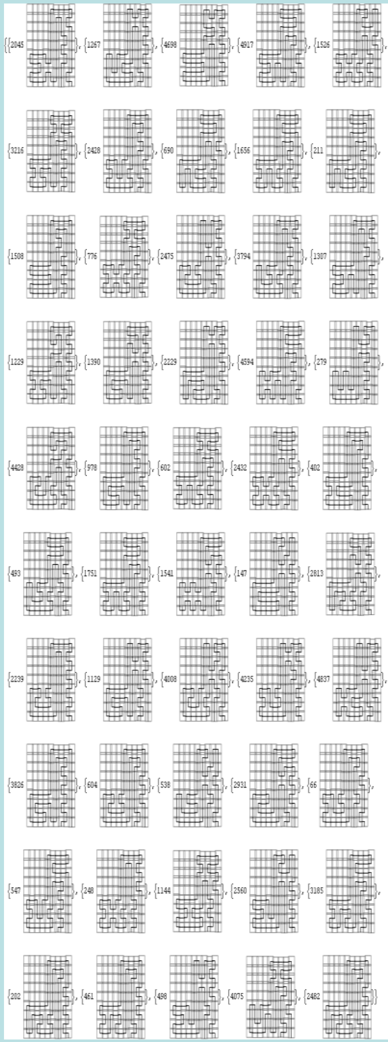
- From the creative space, only accept architectures into the solution space that:
  - Connect 2 pipes from Zone 1 and 2
  - Connect 2 pipes from Zone 1 and 3
  - Have highest counts of straight length pipes (versus bent pipes) -- least action:
    - Less pumping power
    - Lowest cost
      - Lower pipe cost
      - Lower installation cost

Stage 5 – Configuring the Complete Underfloor Heating Supersystem Containing 3 Modules (zones) Addressing a Given Specification

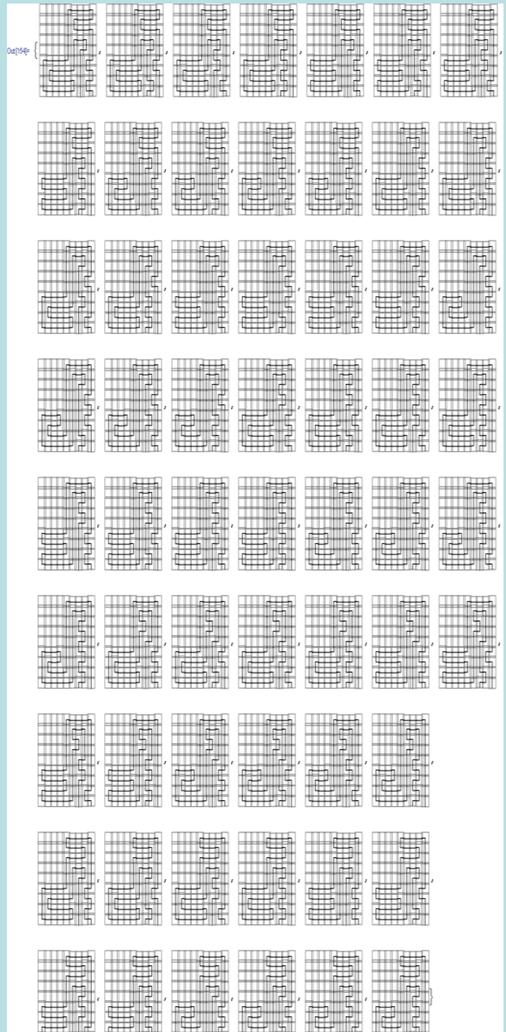


Stages 6 –
Selecting from the Solution Space

50 Samples Randomly taken from the 5032 Piping Solutions, along with their Identification Sequence Number



Least Action Group with Grid, Markers Erased, 60 solutions



Least Action Group without Grid, Markers Erased, 60 solutions