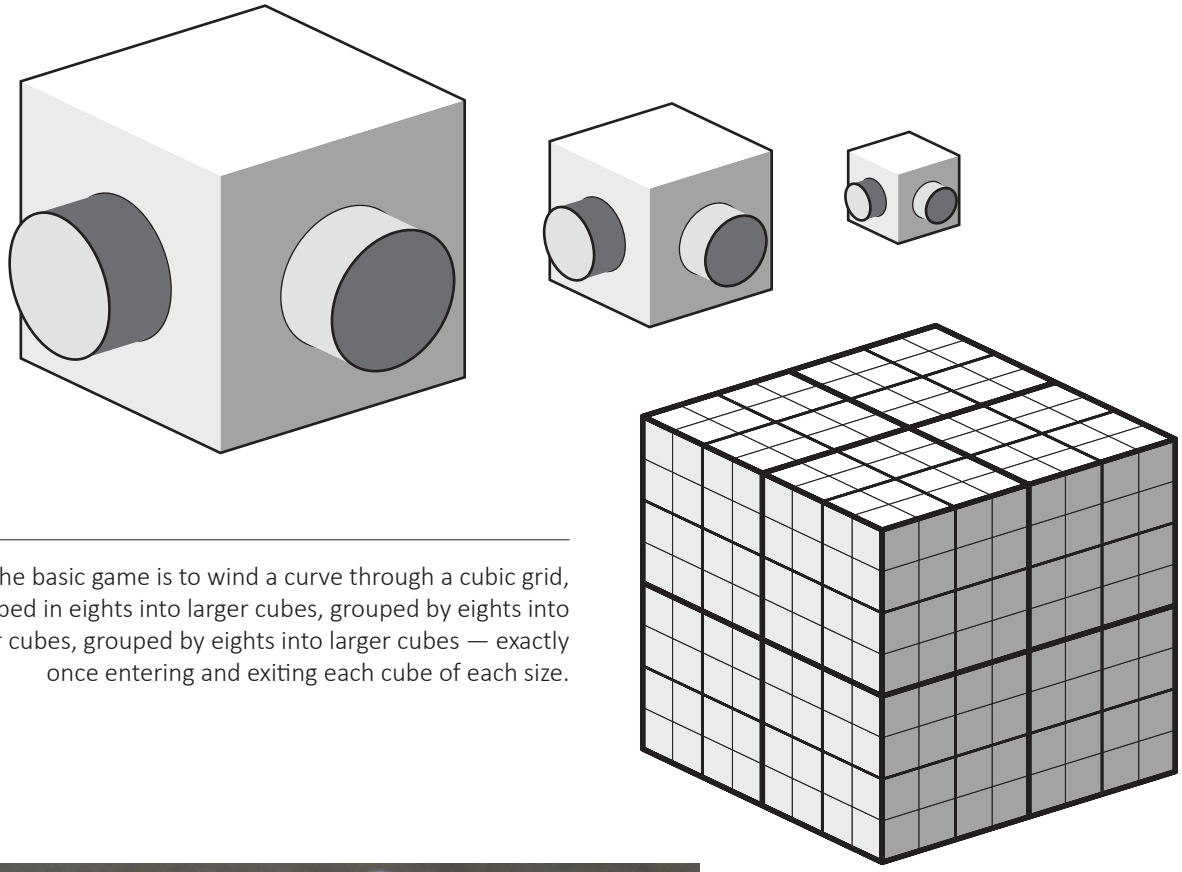
1 of N

Chaim Goodman-Strauss and Eugene Sargent

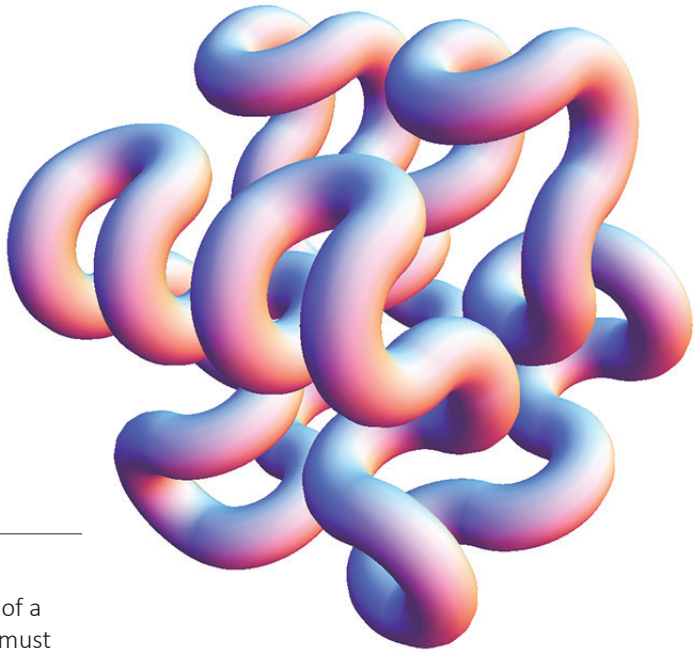
for G4G9 Atlanta, March 2010



The basic game is to wind a curve through a cubic grid, grouped in eights into larger cubes, grouped by eights into larger cubes, grouped by eights into larger cubes — exactly once entering and exiting each cube of each size.

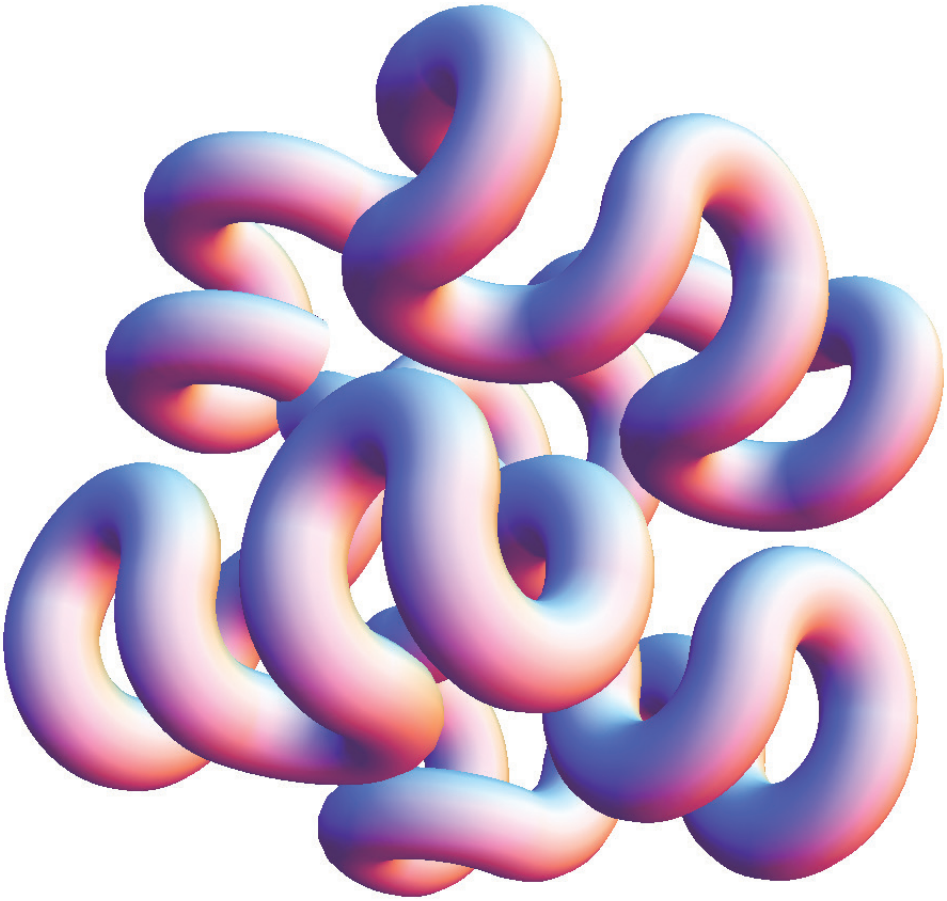
But up to symmetry, how many such curves are possible?
This number is part of the real name of this sculpture.
We have left this calculation as a puzzle, and in the meantime, call the sculpture 1 of N.

We made a quick model out of sixty-four pieces of plumbing.

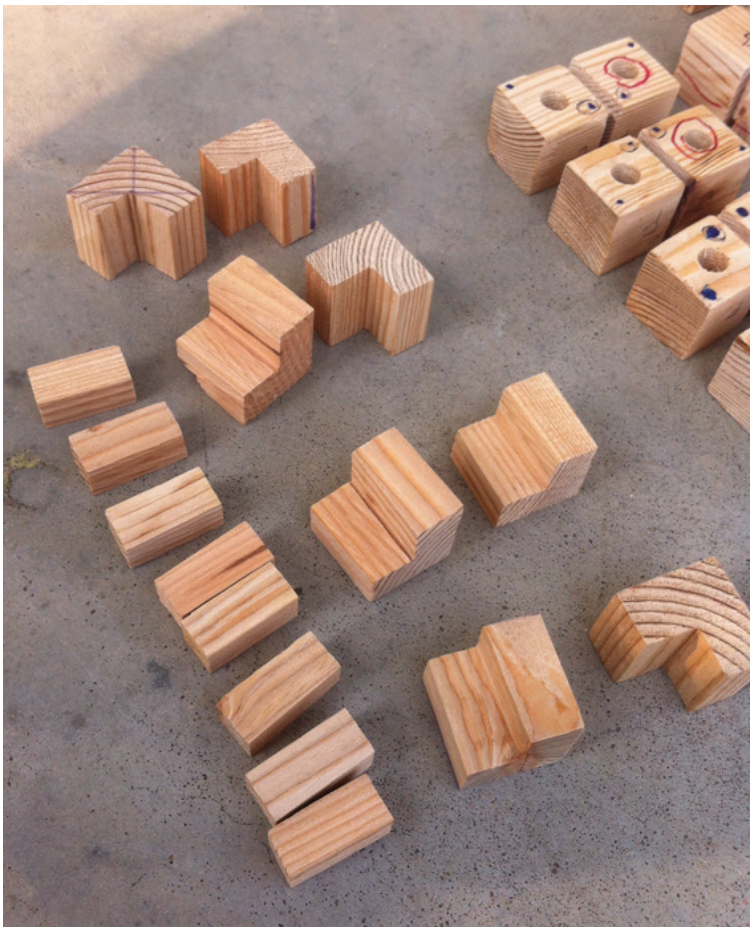


The task would be to count the ways of recursively generating sucn a path; the total number of words of a given length is not difficult to count, but the paths must close—the beginning must match the end.

Moreover, we needed to be able to rummage around the set of all such curves, to try to make some rational choices about the final form of the sculpture.



(Chaim's formalism of "Regular Production Systems" are designed to capture just this sort of non-deterministic production system acting under regular constraints, arising in a wide variety of settings.)



Alas, I cannot find the folder with extensive hand-written notes and diagrams developing the structure.

From these, models were made and code was written.

This first set of wooden tiles consists of eight bricks, and eight clumps of three bricks, in the shape of an L.

It is not difficult to enumerate all arrangements of these, up to symmetry, within a 4x4 cube.

Now we want to begin to enumerate all possible ways to substitute in. We presume that the strings represent closed curves. We first encode all "shifts" at the vertices.

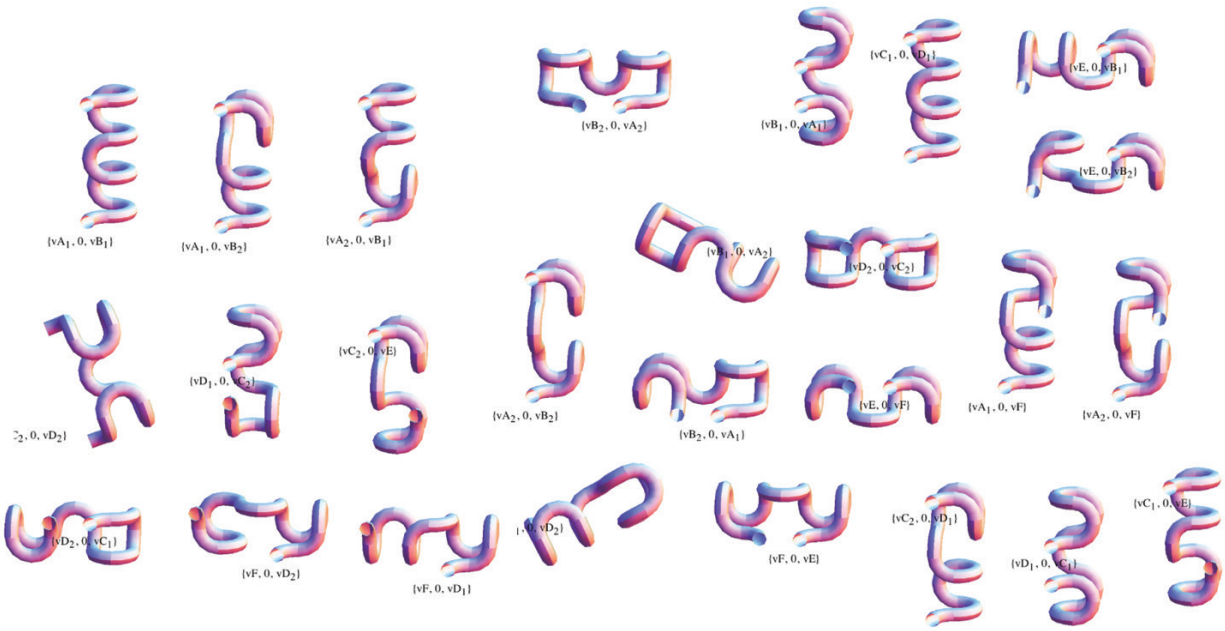
There are six possible ways to shift a vertex (see notes), which for lack of something simpler we represent as

vA, vB, vC, vD, vE, vF

```
Protect[vA, vB, vC, vD, vE, vF]

{vA, vB, vC, vD, vE, vF}

NextShiftCanBe[edgeSymbol_, lastShift_] := Switch[edgeSymbol,
  0, Which[
   MemberQ[{vA1, vA2, vE}, lastShift], {vB1, vB2, vF},
   MemberQ[{vB1, vB2}, lastShift], {vA1, vA2},
   MemberQ[{vC1, vC2, vF}, lastShift], {vD1, vD2, vE},
   MemberQ[{vD1, vD2}, lastShift], {vC1, vC2},
   True, Print["ERROR 0"]],
  1, Which[
   MemberQ[{vA1, vA2, vE}, lastShift], {vD1, vD2, vE},
   MemberQ[{vB1, vB2}, lastShift], {vB1, vB2, vF},
   MemberQ[{vC1, vC2, vF}, lastShift], {vC1, vC2},
   MemberQ[{vD1, vD2}, lastShift], {vA1, vA2},
   True, Print["ERROR 1"]],
  2,

  CanonicalSubstitution[str_] := (
      ss = {vA1};
      sstr = str;
      While[Length[sstr] > 0,
       ee = sstr[[1]];
       sstr = Drop[sstr, 1];
       ss = Join[ss, (ee, NextShiftCanBe[ee, ss[[-1]][[1]])]];
      ss
     ) // (xx = Drop[AdjustEdges[yy = #], -1]) & // CleanoutHilbertTempString
  RandomSubstitution[str_] := (
      ss = RandomChoice[{#} & /@ {vA1, vB1, vC1, vD1, vA2, vB2, vC2, vD2, vE, vF}];
      sstr = str;
      While[Length[sstr] > 0,
       ee = sstr[[1]];
       sstr = Drop[sstr, 1];
       ss = Join[ss, (ee, RandomChoice[NextShiftCanBe[ee, ss[[-1]]]])];
      ss
     ) // (xx = Drop[AdjustEdges[yy = #], -1]) & // CleanoutHilbertTempString

  (*these are our starting strings*)
  sstr = str;
  While[Length[sstr] > 0,
   ee = sstr[[1]];
   sstr = Drop[sstr, 1];
   ss = Join @@ (Join @@ (Outer[Append, {#[[1]]}, #[[2]], 1] & /@
         ({Append[#, ee], NextShiftCanBe[ee, #[[-1]]]} & /@ ss)));

CanonicalSubstitution[str_] := (
    ss = {vA1};
    sstr = str;
    While[Length[sstr] > 0,
     ee = sstr[[1]];
     sstr = Drop[sstr, 1];
     ss = Join[ss, (ee, NextShiftCanBe[ee, ss[[-1]][[1]])]];
    ss
   ) // (xx = Drop[AdjustEdges[yy = #], -1]) & // CleanoutHilbertTempString
RandomSubstitution[str_] := (
    ss = RandomChoice[{#} & /@ {vA1, vB1, vC1, vD1, vA2, vB2, vC2, vD2, vE, vF}];
    sstr = str;
    While[Length[sstr] > 0,
     ee = sstr[[1]];
     sstr = Drop[sstr, 1];
     ss = Join[ss, (ee, RandomChoice[NextShiftCanBe[ee, ss[[-1]]]])];
    ss
   ) // (xx = Drop[AdjustEdges[yy = #], -1]) & // CleanoutHilbertTempString
```

$$
\begin{array}{llllllll}
\{\{vE, 0, vB_1\}, & \{1\}, & \{vB_1, 1, vB_1\}, & \{0\}, \\
\{vA_1, 3, vA_2\}, & \{0\}, & \{vB_2, 1, vF\}, & \{2\}, \\
\{vA_1, 0, vF\}, & \{1\}, & \{vC_2, 1, vC_1\}, & \{0\}, \\
\{vD_2, 3, vE\}, & \{0\}, & \{vB_1, 1, vF\}, & \{2\}, \\
\{vA_2, 1, vD_2\}, & \{3\}, & \{vD_1, 0, vC_2\}, & \{3\}, \\
\{vB_1, 0, vA_2\}, & \{1\}, & \{vD_2, 0, vC_1\}, & \{2\}, \\
\{vA_1, 1, vD_2\}, & \{3\}, & \{vE, 0, vB_1\}, & \{3\}, \\
\{vC_2, 0, vD_1\}, & \{1\}, & \{vA_2, 0, vB_2\}, & \{2\}, \\
\{vD_1, 0, vC_1\}, & \{1\}, & \{vC_1, 1, vC_2\}, & \{0\}, \\
\{vE, 3, vA_1\}, & \{0\}, & \{vF, 1, vC_1\}, & \{2\}, \\
\{vA_1, 0, vB_2\}, & \{2\}, & \{vE, 0, vB_2\}, & \{3\}, \\
\{vC_1, 0, vD_1\}, & \{1\}, & \{vA_2, 0, vB_1\}, & \{1\}, \\
\{vF, 0, vE\}, & \{3\}, & \{vA_1, 0, vB_1\}, & \{3\}, \\
\{vC_2, 0, vE\}, & \{1\}, & \{vD_2, 0, vC_2\}, & \{2\}, \\
\{vA_2, 0, vF\}, & \{2\}, & \{vA_1, 3, vA_1\}, & \{0\}, \\
\{vF, 1, vC_2\}, & \{0\}, & \{vD_2, 3, vD_1\}\}
\end{array}
$$

These are the individual bricks themselves, and code to enumerate them.



In turn, arrangements of these bricks were encoded on another set of blocks, so that the system could be explored by hand.

. . . leading to this speciman, one of N possibilities.

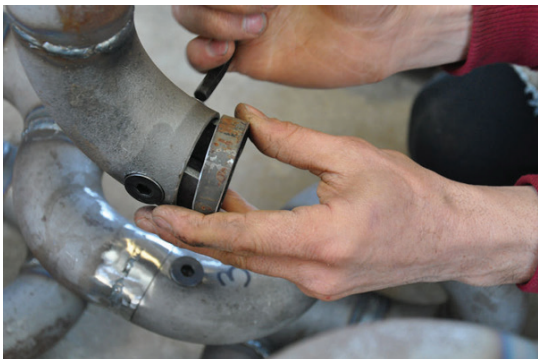
Chaim sent Eugene the following message:

```
033010330110301102010303120103O 313301033011030120230103301103011
320301013110301103301032020103O 312103011033010331303010201103011
311030121210301131010302033010 3 203030102011030121210301102010303
020301010210301103301033130301O 213301032310103023110301202301033
010103020201030312010303133010 3 203301033020301013110301102010303
020301013230103313301033023010 3 312010303010103023110301213030102
011030110330103302030101030301O 213301032033010331210301103301032
311030120110301132030101320301O 131103011033010320210301131010302
```

Each digit encoded a turn from one module to the next: 0 for none, 1 for a quarter turn, 2 for a half turn and 3 for a quarter turn the other way.

Eugene made special jigs, one for each digit, so the seams could be smoothly welded.

All together, each block of fifteen digits encoded a brick made up of sixteen modules.



Thurman Thomas and Lee Koehn.

There were thirty-two bricks, each with a unique place in the sculpture.

The individual pieces weren't very uniform—next time we will use artist-grade plumbing.

As always, assembling the piece at the party is the best part of all.

Unfortunately, we'd left our carefully prepared notes back in the hotel and had to work out how to put it together again!