Analysis of Conway - Piccirillo Knot with Turing State Machine

Lisa Piccirillo - Solving the Conway Knot

https://en.wikipedia.org/wiki/Lisa_Piccirillo

Matt Parker - Stand-Up Maths

http://standupmaths.com/

Hypercube and Turing Machine Videos

Moritz Firsching - Unfolding of the Hypercube

https://unfolding.apperceptual.com/

Pascal Michel - Busy Beaver Game (Explanation)

https://webusers.imj-prg.fr/~pascal.michel/

The inspiration for this paper came from thinking about whether the same principle employed within the "Analysis of hypercube tree topology" could be applied to another scenario. Available at the below link.

https://github.com/Sandcrawler/hypercube

I thought about what else could be represented as a tree diagram, and thought knots are tree diagrams that are attached at both ends. We can break the end and halt it (as we do in the previous paper) and look to see what the pattern looks like.

To begin, I had to choose a knot. I had watched a documentary recently, with a section covering the Conway Knot and the solution that Lisa Picirillo gave for it. This seemed a good place to start.

This paper covers representing the Picirillo Knot as a sigma(11,5) state machine.

The Knots

We have two knots. One that had been presented as being unsolvable and one that is a solution to the apparently unsolvable.

Let us take a look at how we can map these onto a state machine.

Conway Knot

Consider the knot as having 11 intersections.

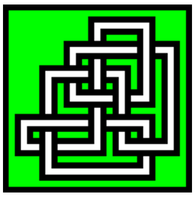

Figure 1.1

No state machine can be built to satisfy an 11-node tree.

Note: No state machine can be built for any prime number of node trees.

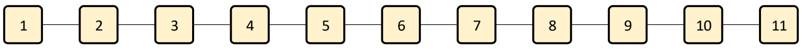

Figure 1.2

Picirillo Knot

Consider the knot as having 55 intersections.

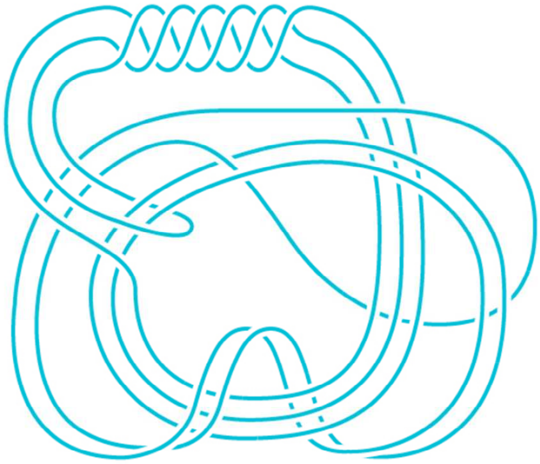

Figure 2.1

There is a state machine matrix available at 55-node that is symmetrical across both X and Y planes.

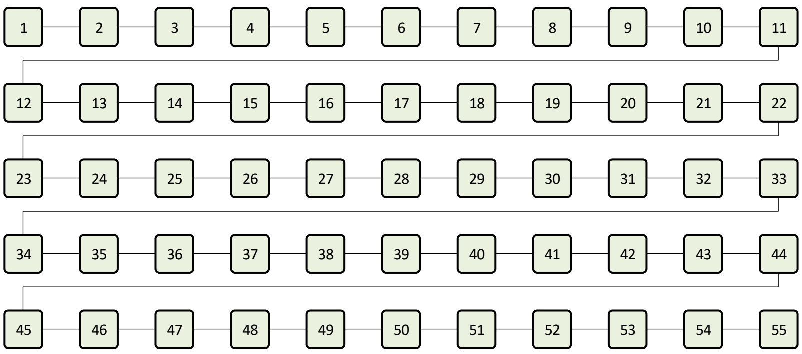

Figure 2.2

This is the symmetrical state machine:

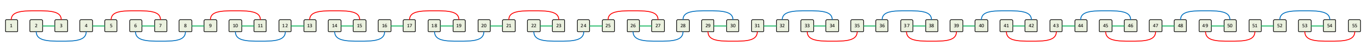

Figure 2.3

In the previous paper “Analysis of a hypercube tree topology” we rotate the state machine around every iteration of the machine to demonstrate that there are only 2 potentials to use and why one of them is viable. In this paper we will look at just the symmetrical tree and why the 55 intersections works rather than the 11. Suggest reading the previous paper to understand the rationalisation of the tree.

The following diagram shows the path that the state machine must follow to achieve every node and pass through each only once within a symmetrical manner.

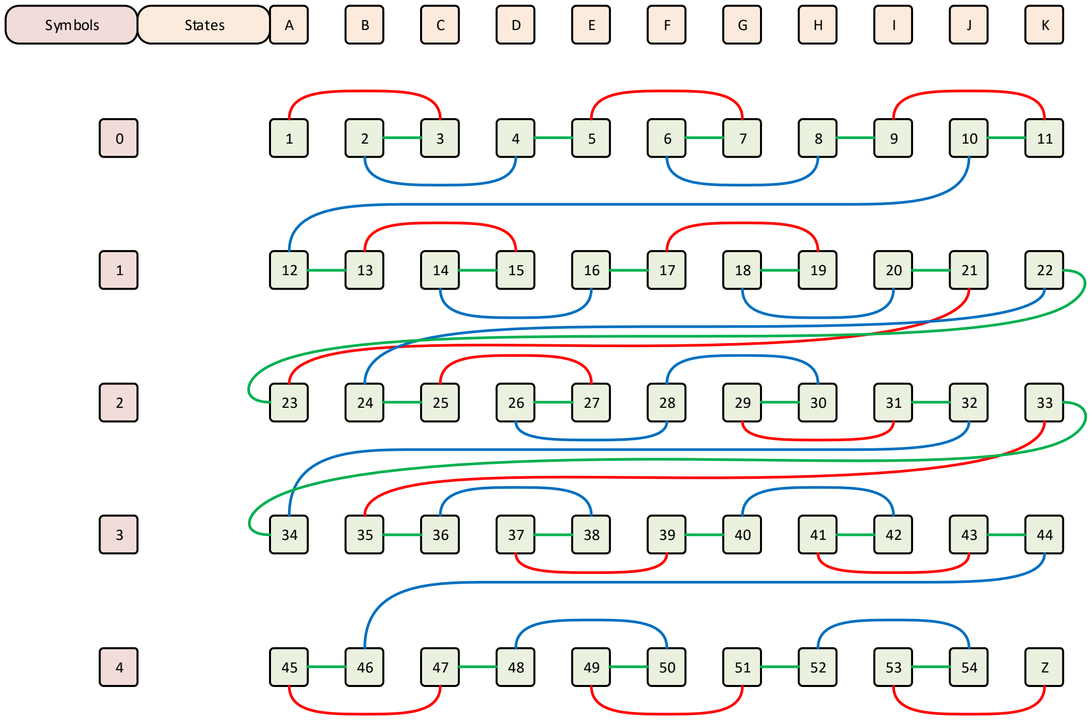
Symbols
States
A
B
C
D
E
F
G
H
I
J
K
0
1
2
3
4
1 2 3 4 5 6 7 8 9 10 11
12 13 14 15 16 17 18 19 20 21 22
23 24 25 26 27 28 29 30 31 32 33
34 35 36 37 38 39 40 41 42 43 44
45 46 47 48 49 50 51 52 53 54 Z

Figure 2.4

Constructing a State Machine

One potential forward and inverse state machines of the Picirillo Knot.

		Array Pointer			Array Pointer				
	0LA	0	0	0	0	0	0	0RA	
A0	0LC		0	0	0	0		0RC	A0
C0	0RB	0	0			0	0	0LB	C0
B0	0LD		0	0	0	0		0RD	B0
D0	0RE	0	0			0	0	0LE	D0
E0	0LG		0	0	0	0		0RG	E0
G0	0RF	0	0			0	0	0LF	G0
F0	0LH		0	0	0	0		0RH	F0
H0	0RI	0	0			0	0	0LI	H0
I0	0LK		0	0	0	0		0RK	I0
K0	1RJ	1	0			0	1	1LJ	K0
J0	1LA		1	1	1	1		1RA	J0
A1	1RB	1	1			1	1	1LB	A1
B1	1LD		1	1	1	1		1RD	B1
D1	1RC	1	1			1	1	1LC	D1
C1	1LE		1	1	1	1		1RE	C1
E1	1RF	1	1			1	1	1LF	E1
F1	1LH		1	1	1	1		1RH	F1
H1	1RG	1	1			1	1	1LG	H1
G1	1LI		1	1	1	1		1RI	G1
I1	2RJ	2	1			1	2	2LJ	I1
J1	1LA		2	2	2	2		2RA	J1
A2	2RK	2	2			2	2	2LK	A2
K1	2LB		2	2	2	2		2RB	K1
B2	2RC	2	2			2	2	2LC	B2
C2	2LE		2	2	2	2		2RE	C2
E2	2RD	2	2			2	2	2LD	E2
D2	2LF		2	2	2	2		2RF	D2
F2	2RH	2	2			2	2	2LH	F2
H2	2LG		2	2	2	2		2RG	H2
G2	2RI	2	2			2	2	2LI	G2
I2	3LJ		2	2	2	2		3RJ	I2
J2	2RA	2	3			3	2	2LA	J2
A3	3LK		3	3	3	3		3RK	A3
K2	3RB	3	3			3	3	3LB	K2
B3	3LC		3	3	3	3		3RC	B3
C3	3RE	3	3			3	3	3LE	C3
E3	3LD		3	3	3	3		3RD	E3
D3	3RF	3	3			3	3	3LF	D3
F3	3LG		3	3	3	3		3RG	F3
G3	3RI	3	3			3	3	3LI	G3
I3	3LH		3	3	3	3		3RH	I3
H3	3RJ	3	3			3	3	3LJ	H3
J3	4LK		4	3	3	4		4RK	J3
K3	4RB	4	4			4	4	4LB	K3
B4	4LA		4	4	4	4		4RA	B4
A4	4RC	4	4			4	4	4LC	A4
C4	4LD		4	4	4	4		4RD	C4
D4	4RF	4	4			4	4	4LF	D4
F4	4LE		4	4	4	4		4RE	F4
E4	4RG	4	4			4	4	4LG	E4
G4	4LH		4	4	4	4		4RH	G4
H4	4RJ	4	4			4	4	4LJ	H4
J4	4LI		4	4	4	4		4RI	J4
I4	4RK	4	4			4	4	4LK	I4
K4 (Z)	4LZ		4	4	4	4		4RZ	K4 (Z)

Figure 3.1

Following the same logic as the previous paper this is the inverse of the above. We now have all 4 answers to the same state machine and we can follow the same pattern to merge them. Un-invert the inverted result and combine. The un-inverted result is shown on the right.

		Array Pointer			Array Pointer				
	0RA	0	0	0	0	0	0	0LA	
A0	0RC	0	0			0	0	0LC	A0
C0	0LB		0	0	0	0		0RB	C0
B0	0RD	0	0			0	0	0LD	B0
D0	0LE		0	0	0	0		0RE	D0
E0	0RG	0	0			0	0	0LG	E0
G0	0LF		0	0	0	0		0RF	G0
F0	0RH	0	0			0	0	0LH	F0
H0	0LI		0	0	0	0		0RI	H0
I0	0RK	0	0			0	0	0LK	I0
K0	1LJ		0	1	1	0		1RJ	K0
J0	1RA	1	1			1	1	1LA	J0
A1	1LB		1	1	1	1		1RB	A1
B1	1RD	1	1			1	1	1LD	B1
D1	1LC		1	1	1	1		1RC	D1
C1	1RE	1	1			1	1	1LE	C1
E1	1LF		1	1	1	1		1RF	E1
F1	1RH	1	1			1	1	1LH	F1
H1	1LG		1	1	1	1		1RG	H1
G1	1RI	1	1			1	1	1LI	G1
I1	2LJ		1	2	2	1		2RJ	I1
J1	2RA	2	2			2	2	1LA	J1
A2	2LK		2	2	2	2		2RK	A2
K1	2RB	2	2			2	2	2LB	K1
B2	2LC		2	2	2	2		2RC	B2
C2	2RE	2	2			2	2	2LE	C2
E2	2LD		2	2	2	2		2RD	E2
D2	2RF	2	2			2	2	2LF	D2
F2	2LH		2	2	2	2		2RH	F2
H2	2RG	2	2			2	2	2LG	H2
G2	2LI		2	2	2	2		2RI	G2
I2	3RJ	2	2			2	2	3LJ	I2
J2	2LA		3	2	2	3		2RA	J2
A3	3RK	3	3			3	3	3LK	A3
K2	3LB		3	3	3	3		3RB	K2
B3	3RC	3	3			3	3	3LC	B3
C3	3LE		3	3	3	3		3RE	C3
E3	3RD	3	3			3	3	3LD	E3
D3	3LF		3	3	3	3		3RF	D3
F3	3RG	3	3			3	3	3LG	F3
G3	3LI		3	3	3	3		3RI	G3
I3	3RH	3	3			3	3	3LH	I3
H3	3LJ		3	3	3	3		3RJ	H3
J3	4RK	3	4			4	3	4LK	J3
K3	4LB		4	4	4	4		4RB	K3
B4	4RA	4	4			4	4	4LA	B4
A4	4LC		4	4	4	4		4RC	A4
C4	4RD	4	4			4	4	4LD	C4
D4	4LF		4	4	4	4		4RF	D4
F4	4RE	4	4			4	4	4LE	F4
E4	4LG		4	4	4	4		4RG	E4
G4	4RH	4	4			4	4	4LH	G4
H4	4LJ		4	4	4	4		4RJ	H4
J4	4RI	4	4			4	4	4LI	J4
I4	4LK		4	4	4	4		4RK	I4
K4 (Z)	4RZ	4	4			4	4	4LZ	K4 (Z)

4	4			4	4
	4	4	4	4	
4	4			4	4
	4	4	4	4	
4	4			4	4
	4	4	4	4	
4	4			4	4
	4	4	4	4	
4	4			4	4
	4	4	4	4	
4	4			4	4
	4	4	4	4	
3	4			4	3
	3	3	3	3	
3	3			3	3
	3	3	3	3	
3	3			3	3
	3	3	3	3	
3	3			3	3
	3	3	3	3	
3	3			3	3
	3	3	3	3	
3	3			3	3
	3	2	2	3	
2	2			2	2
	2	2	2	2	
2	2			2	2
	2	2	2	2	
2	2			2	2
	2	2	2	2	
2	2			2	2
	2	2	2	2	
2	2			2	2
	2	2	2	2	
2	2			2	2
	1	2	2	1	
1	1			1	1
	1	1	1	1	
1	1			1	1
	1	1	1	1	
1	1			1	1
	1	1	1	1	
1	1			1	1
	1	1	1	1	
1	1			1	1
	0	1	1	0	
0	0			0	0
	0	0	0	0	
0	0			0	0
	0	0	0	0	
0	0			0	0
	0	0	0	0	
0	0			0	0
	0	0	0	0	
0	0			0	0

Figure 3.2 and 3.3

Combining the Matrix

We take the result of the first and then the X and Y flip of the second to un-invert it. We then combine the matrix.

Forward + Un-Inverted = Resultant Matrix

Forward

0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
1	0	1
1	2	1
1	2	1
1	2	1
1	2	1
1	2	1
1	2	1
1	2	1
1	2	1
1	2	1
2	2	2
2	4	2
2	4	2
2	4	2
2	4	2
2	4	2
2	4	2
2	4	2
2	4	2
2	4	2
2	4	2
2	4	2
2	6	2
3	6	3
3	6	3
3	6	3
3	6	3
3	6	3
3	6	3
3	6	3
3	6	3
3	6	3
3	6	3
3	8	3
4	8	4
4	8	4
4	8	4
4	8	4
4	8	4
4	8	4
4	8	4
4	8	4
4	8	4
4	8	4
4	8	4
4	8	4

Un-Inverted

4	8	4
4	8	4
4	8	4
4	8	4
4	8	4
4	8	4
4	8	4
4	8	4
4	8	4
4	8	4
4	8	4
4	8	4
3	8	3
3	6	3
3	6	3
3	6	3
3	6	3
3	6	3
3	6	3
3	6	3
3	6	3
3	6	3
3	6	3
2	6	2
2	4	2
2	4	2
2	4	2
2	4	2
2	4	2
2	4	2
2	4	2
2	4	2
2	4	2
2	4	2
2	4	2
2	2	2
1	2	1
1	2	1
1	2	1
1	2	1
1	2	1
1	2	1
1	2	1
1	2	1
1	2	1
1	0	1
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0

Resultant Matrix

	0LA						0RA	
A0	0LC	0	4	8	4	0	0RC	A0
C0	0RB	0	4	8	4	0	0LB	C0
B0	0LD	0	4	8	4	0	0RD	B0
D0	0RE	0	4	8	4	0	0LE	D0
E0	0LG	0	4	8	4	0	0RG	E0
G0	0RF	0	4	8	4	0	0LF	G0
F0	0LH	0	4	8	4	0	0RH	F0
H0	0RI	0	4	8	4	0	0LI	H0
I0	0LK	0	4	8	4	0	0RK	I0
K0	1RJ	0	5	8	5	0	1LJ	K0
J0	1LA	0	5	10	5	0	1RA	J0
A1	1RB	0	5	10	5	0	1LB	A1
B1	1LD	0	4	10	4	0	1RD	B1
D1	1RC	0	4	8	4	0	1LC	D1
C1	1LE	0	4	8	4	0	1RE	C1
E1	1RF	0	4	8	4	0	1LF	E1
F1	1LH	0	4	8	4	0	1RH	F1
H1	1RG	0	4	8	4	0	1LG	H1
G1	1LI	0	4	8	4	0	1RI	G1
I1	2RJ	0	5	8	5	0	2LJ	I1
J1	1LA	0	5	10	5	0	2RA	J1
A2	2RK	0	5	10	5	0	2LK	A2
K1	2LB	0	5	10	5	0	2RB	K1
B2	2RC	0	4	10	4	0	2LC	B2
C2	2LE	0	4	8	4	0	2RE	C2
E2	2RD	0	4	8	4	0	2LD	E2
D2	2LF	0	4	8	4	0	2RF	D2
F2	2RH	0	4	8	4	0	2LH	F2
H2	2LG	0	4	8	4	0	2RG	H2
G2	2RI	0	4	8	4	0	2LI	G2
I2	3LJ	0	4	8	4	0	3RJ	I2
J2	2RA	0	4	10	4	0	2LA	J2
A3	3LK	0	5	10	5	0	3RK	A3
K2	3RB	0	5	10	5	0	3LB	K2
B3	3LC	0	5	10	5	0	3RC	B3
C3	3RE	0	5	8	5	0	3LE	C3
E3	3LD	0	4	8	4	0	3RD	E3
D3	3RF	0	4	8	4	0	3LF	D3
F3	3LG	0	4	8	4	0	3RG	F3
G3	3RI	0	4	8	4	0	3LI	G3
I3	3LH	0	4	8	4	0	3RH	I3
H3	3RJ	0	4	8	4	0	3LJ	H3
J3	4LK	0	4	10	4	0	4RK	J3
K3	4RB	0	5	10	5	0	4LB	K3
B4	4LA	0	5	10	5	0	4RA	B4
A4	4RC	0	5	8	5	0	4LC	A4
C4	4LD	0	4	8	4	0	4RD	C4
D4	4RF	0	4	8	4	0	4LF	D4
F4	4LE	0	4	8	4	0	4RE	F4
E4	4RG	0	4	8	4	0	4LG	E4
G4	4LH	0	4	8	4	0	4RH	G4
H4	4RJ	0	4	8	4	0	4LJ	H4
J4	4LI	0	4	8	4	0	4RI	J4
I4	4RK	0	4	8	4	0	4LK	I4
K4 (Z)	4LZ	0	4	8	4	0	4RZ	K4 (Z)

Figure 4.1

Result

	State Machine Instructions								State Machine Instructions	
		0LA						0RA		
Entry	A0	0LC	0	4	8	4	0	0RC	A0	Exit
	C0	0RB	0	4	8	4	0	0LB	C0	
	B0	0LD	0	4	8	4	0	0RD	B0	
	D0	0RE	0	4	8	4	0	0LE	D0	
	E0	0LG	0	4	8	4	0	0RG	E0	
	G0	0RF	0	4	8	4	0	0LF	G0	
	F0	0LH	0	4	8	4	0	0RH	F0	
	H0	0RI	0	4	8	4	0	0LI	H0	
	I0	0LK	0	4	8	4	0	0RK	I0	
	K0	1RJ	0	5	8	5	0	1LJ	K0	
	J0	1LA	0	5	10	5	0	1RA	J0	
	A1	1RB	0	5	10	5	0	1LB	A1	
	B1	1LD	0	4	10	4	0	1RD	B1	
	D1	1RC	0	4	8	4	0	1LC	D1	
	C1	1LE	0	4	8	4	0	1RE	C1	
	E1	1RF	0	4	8	4	0	1LF	E1	
	F1	1LH	0	4	8	4	0	1RH	F1	
	H1	1RG	0	4	8	4	0	1LG	H1	
	G1	1LI	0	4	8	4	0	1RI	G1	
	I1	2RJ	0	5	8	5	0	2LJ	I1	
	J1	1LA	0	5	10	5	0	2RA	J1	
	A2	2RK	0	5	10	5	0	2LK	A2	
	K1	2LB	0	5	10	5	0	2RB	K1	
	B2	2RC	0	4	10	4	0	2LC	B2	
	C2	2LE	0	4	8	4	0	2RE	C2	
	E2	2RD	0	4	8	4	0	2LD	E2	
	D2	2LF	0	4	8	4	0	2RF	D2	
	F2	2RH	0	4	8	4	0	2LH	F2	
	H2	2LG	0	4	8	4	0	2RG	H2	
	G2	2RI	0	4	8	4	0	2LI	G2	
	I2	3LJ	0	4	8	4	0	3RJ	I2	
	J2	2RA	0	4	10	4	0	2LA	J2	
	A3	3LK	0	5	10	5	0	3RK	A3	
	K2	3RB	0	5	10	5	0	3LB	K2	
	B3	3LC	0	5	10	5	0	3RC	B3	
	C3	3RE	0	5	8	5	0	3LE	C3	
	E3	3LD	0	4	8	4	0	3RD	E3	
	D3	3RF	0	4	8	4	0	3LF	D3	
	F3	3LG	0	4	8	4	0	3RG	F3	
	G3	3RI	0	4	8	4	0	3LI	G3	
	I3	3LH	0	4	8	4	0	3RH	I3	
	H3	3RJ	0	4	8	4	0	3LJ	H3	
	J3	4LK	0	4	10	4	0	4RK	J3	
	K3	4RB	0	5	10	5	0	4LB	K3	
	B4	4LA	0	5	10	5	0	4RA	B4	
	A4	4RC	0	5	8	5	0	4LC	A4	
	C4	4LD	0	4	8	4	0	4RD	C4	
	D4	4RF	0	4	8	4	0	4LF	D4	
	F4	4LE	0	4	8	4	0	4RE	F4	
	E4	4RG	0	4	8	4	0	4LG	E4	
	G4	4LH	0	4	8	4	0	4RH	G4	
	H4	4RJ	0	4	8	4	0	4LJ	H4	
	J4	4LI	0	4	8	4	0	4RI	J4	
	I4	4RK	0	4	8	4	0	4LK	I4	
Exit	K4 (Z)	4LZ	0	4	8	4	0	4RZ	K4 (Z)	Entry

Figure 5.1

Observations

There are a range of similar observations to make with the pattern produced by this tree represented as a state machine as in the previous paper.

Here we will limit the observations to the application of the knot topology to the state machine rather than the array shifts and the oscillation of array shifting. It is none the less interesting to observe the comparison.

Observation #1

Conway Knot

This has prime number of intersections, 11, and cannot be represented as a symmetrical state machine (ie; there are not enough nodes to fill a matrix of instructions without adding an imaginary point). Where in previous paper we stated that a sigma(3,3) state machine could be used and an imaginary added optionally, here we must add one to achieve a halting result.

Observation #2

Picirillo Knot

The Picirillo Knot appears to have morphed into a symmetrical number, in that 55 can be represented as a symmetrical state machine through the 2 and F symbol and state axis (X and Y axis). This has the result of enabling a state machine to be developed for it that is perfectly symmetrical.

Observation #3

Knots of prime number joined at the ends are all subject to this same issue. No prime numbered knot can be represented as an X and Y symmetrical state machine.

Observation #4

Prime numbered knots may be able to be built into non-halting state machines.

An 11-node tree could be built into a state machine of sigma(6,2) where it loops around 1-11 never going to halt at node 12.

Observation #5

Choosing what to do at point of halt in machines with many symbols can cause different numbers at what would be the stitching row (see previous paper). In this paper we assume the last iteration is the same symbol as the previous, but it is possible the halting instruction could write a 0,1,2, 3, or 4 before halting altering the values and thus creating a different set of potential results when being reprocessed by the same set of state machine instructions.

It is assumed using symbol 4 in the halt, as if the pattern were to be built into a larger state machine the de facto trajectory of the numbering would, I suspect, be followed incrementally for this knot and state machine.

Observation #6

Based on observation #6. It may be possible at the point of stitching if writing different values on the halt to stitch different shapes together. State machines having an altered value substituted at the stich point would then work with different valued arrays being processed off the end of the stitch.

For example:

Last instruction set in machine: Read value, write value, follow to next instruction which is HALT and so stop immediately.

0, 1, 2, 3, 4 can be written as the final value just before the halt.

Last 4 instructions alternative:

H4	4RJ	4	4			4	4	4LJ	H4
J4	4LI		4	4	4	4		4RI	J4
I4	4RK	4	4			4	4	4LK	I4
K4 (Z)	0LZ		0	4	4	0		0RZ	K4 (Z)

Figure 6.1

The inverse:

H4	4LJ		4	4	4	4		4RJ	H4
J4	4RI	4	4			4	4	4LI	J4
I4	4LK		4	4	4	4		4RK	I4
K4 (Z)	0RZ	4	0			0	4	0LZ	K4 (Z)

Figure 6.2

Combined result would end up as:

Note the 0 in the central pointer track at the last instruction.

This could result in being 0, 2, 4, 6, 8 depending on the last instruction write and would result in a stich going in different potential directions of array values.

	H4	4RJ	0	4	8	4	0	4LJ	H4	
	J4	4LI	0	4	8	4	0	4RI	J4	
	I4	4RK	0	4	8	4	0	4LK	I4	
Exit	K4 (Z)	0LZ	0	4	0	4	0	0RZ	K4 (Z)	Entry

Figure 6.3

Observation #7

If the first 3 instructions were L L L instead of L R L it would yield a different pattern when combing the matrix. Answers with these different shifts at the start where the matrix is 0 would give adjacently running tracks, thus there are seemingly many answers to this particular state machine.

The tracks can be shifted apart by how far the 0 filling of the array will allow at the start.