Analysis of Stacking Sigma Results from a 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/

This paper was inspired by thinking about how to achieve a contra flow of instructions. In previous papers it is assumed that there is an X and Y flip somewhere to achieve the combination of data to produce the results we see.

This paper looks primarily to show how a contra flow set of state machine instructions could be built and overlayed without assuming it is present without reason.

It is recommended to reference previous papers, in particular the Analysis of a Turing Machine to understand the sigma(0,0) result set pattern.

https://github.com/Sandcrawler/turingmachine

Version

Version	
v1	First Draft
v2	Updated Figure 2.1, Figure 4.1 Figure 5.1/5.2 added the tags. Table in quadrant mapping updated. Added section sigma(1,1) Added section Non-Symmetry Resolution Added Figure 6.1, Figure 6.2, Figure 6.3 Added observations about lines.

Sigma(0,0)

In the last paper it was observed that sigma(0,0) could be referenced as sigma(+0,+0), sigma(-0,+0), sigma(+0,-0), sigma(-0,-0).

Let us extract this out further and show how the sigma can be used in a larger pattern set and how this could then allow for a contra flow set of instructions to occur.

Quadrant Representation

Consider one of these quadrants as what we have been working in to now (although with the assumed contra flow of instructions to manifest what the results have been).

Consider a graph with 8 quadrants.

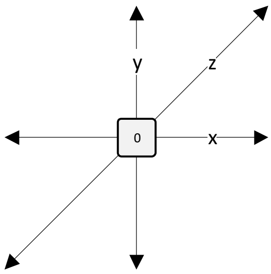

Figure 1.1

Upper Quadrant

sigma(-0,+0) First instruction: LH	sigma(+0,+0) First instruction: RH
sigma(-0-,0) First instruction: RH	sigma(+0,-0) First instruction: LH

Figure 1.2

Lower Quadrant

sigma(-0,+0) First instruction: LH	sigma(+0,+0) First instruction: RH
sigma(-0-,0) First instruction: RH	sigma(+0,-0) First instruction: LH

Figure 1.3

Quadrant Mapping

This diagram shows a simple mapping of quadrants.

Consider 0 as 0,0,0 and each of the 8 quadrant nodes as the infinite opposite of 0,0,0 toward the opposing end of the quadrant.

~~The nodes will then point toward the same infinite points that 0,0,0 will along the axis to create a quadrant.~~

The nodes will then point toward the same corners at the infinite points of X, Y and Z.

Quadrant	**Axis-1**	**Axis-2**	**Axis-3**
TR+	-y	-x	-z
TL+	-y	+x	-z
TR-	-y	-x	+z
TL-	-y	+x	+z
BR+	+y	-x	-z
BL+	+y	+x	-z
BR-	+y	-x	+z
BL-	+y	+x	+z

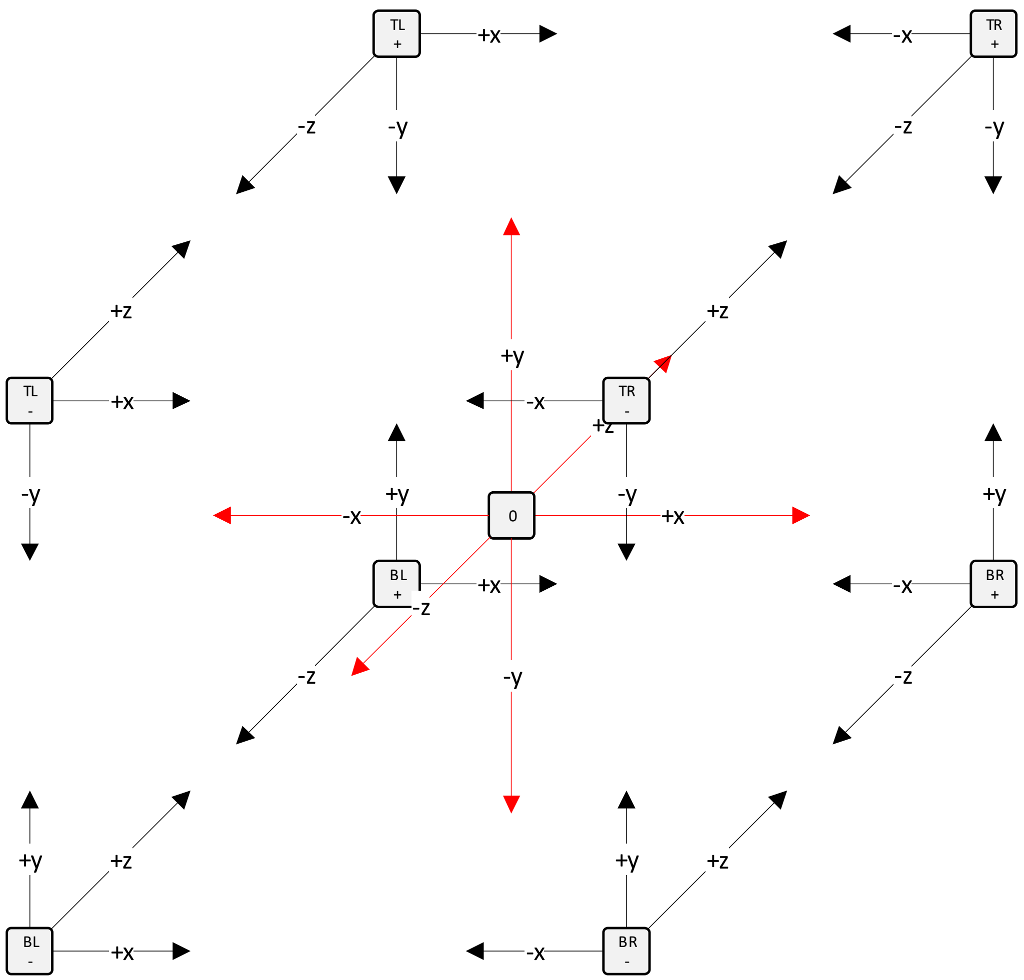

Figure 2.1

Upper Quadrant Mapping

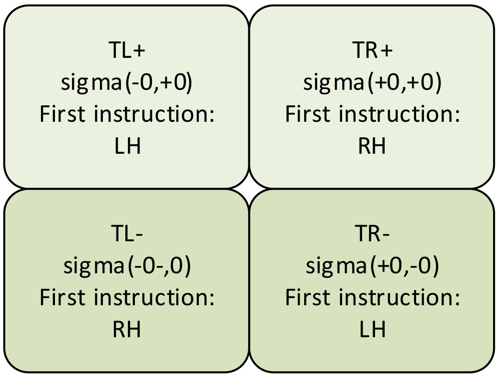

Figure 3.1

Lower Quadrant Mapping

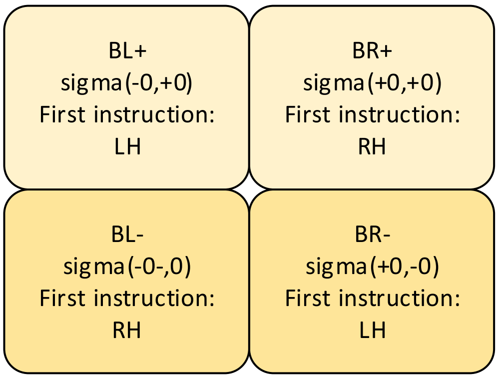

Figure 3.2

We now build the 8-quadrant structure up and overlay the structure on itself in a symmetrical manner by placing 0,0,0 at each of the 8 nodes.

What we see at this point is all 8 quadrants are all equal.

This would suggest that the inverse instruction set is equal to the forward instruction set and that everything starts at 0,0,0 and ends at 0,0,0 and every corner of the quadrant is 0,0,0.

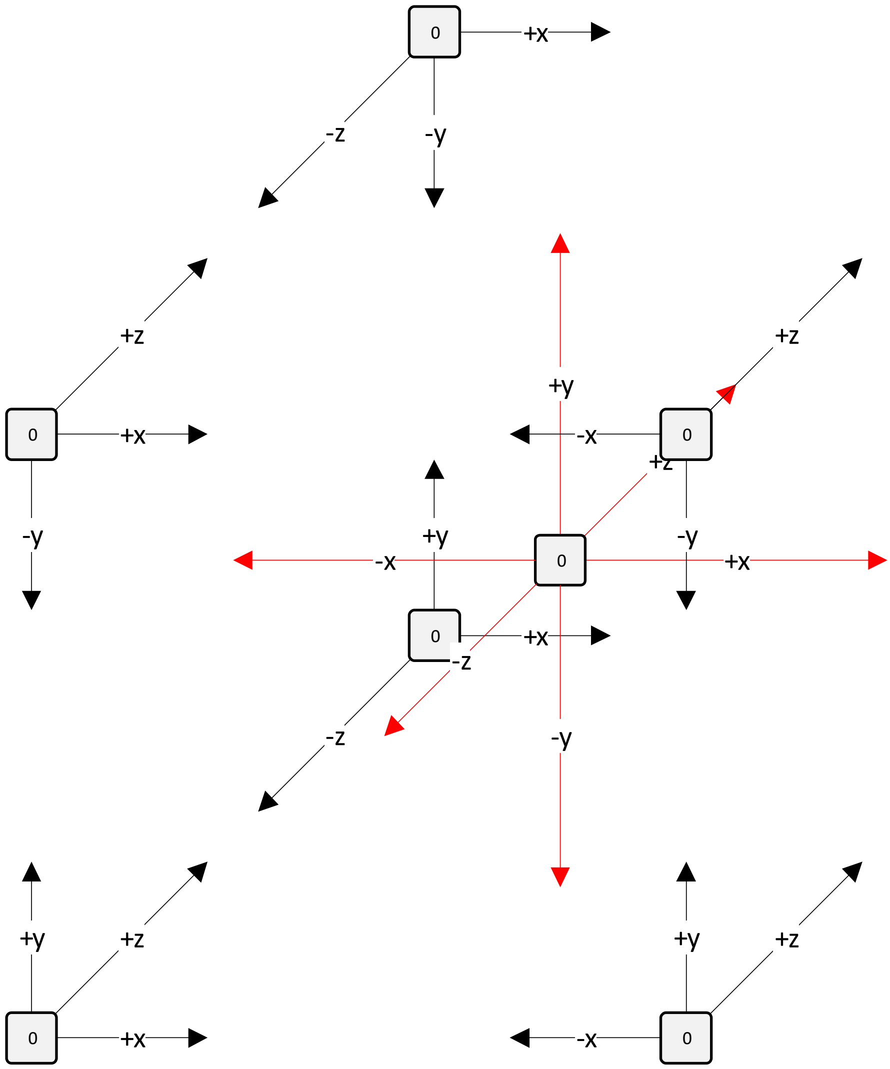

Figure 4.1

State Machine Instruction Flow

Consider from 0,0,0 sigma(+0,+0) runs and follows a trajectory towards TR+

We would see from TR+ (which could be considered as 0,0,0) BL- running toward 0,0,0

This would result in instructions sets in each direction starting with the same Left or Right instruction giving a contra flow set of instructions.

In the simplest example:

LH originating from 0,0,0 -> TR+

LH originating from TR+ -> 0,0,0

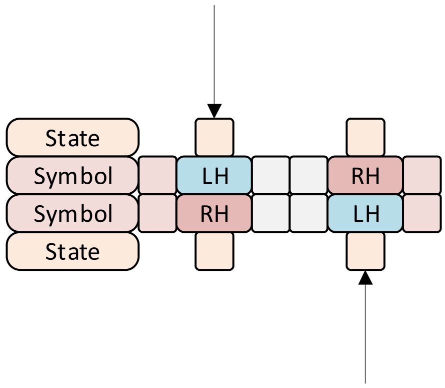

Figure 5.1

If we consider that the quadrant is overlapped with all 8 instructions, we could also suggest the following:

RH originating from 0,0,0 -> TR+

RH originating from TR+ -> 0,0,0

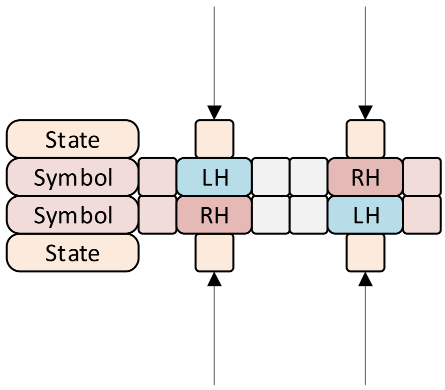

Figure 5.2

Sigma(1,1)

Consider that sigma(0,0) allows 3 dimensional space to exists.

Reference: https://github.com/Sandcrawler/turingmachine

Consider that instruction sets run from sigma(0,0) to sigma(0,0) and that the previous paper shows a linear structure between the two points running contra to each other.

Consider the instruction set to build a circle.

Reference Figure 2.2 in: https://github.com/Sandcrawler/matrices

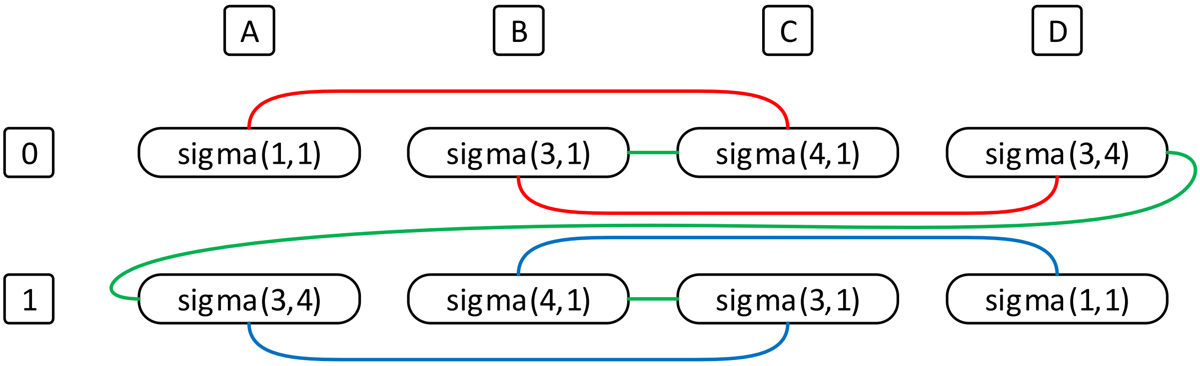

Figure 6.1

Consider the instruction at the start and end of the dot product sigma from the matrix used to calculate Pi. Both are sigma(1,1).

Consider observation #7 in: https://github.com/Sandcrawler/matrices

"**Observation #7**

Is sigma(1,1) an optional HALT?

Is sigma(1,1) the infinite vanishing point for X, Y and Z? Which would make sigma(1,1) = sigma(0,0)?"

If we include sigma(1,1) at the X, Y, and Z vanishing points we get the following.

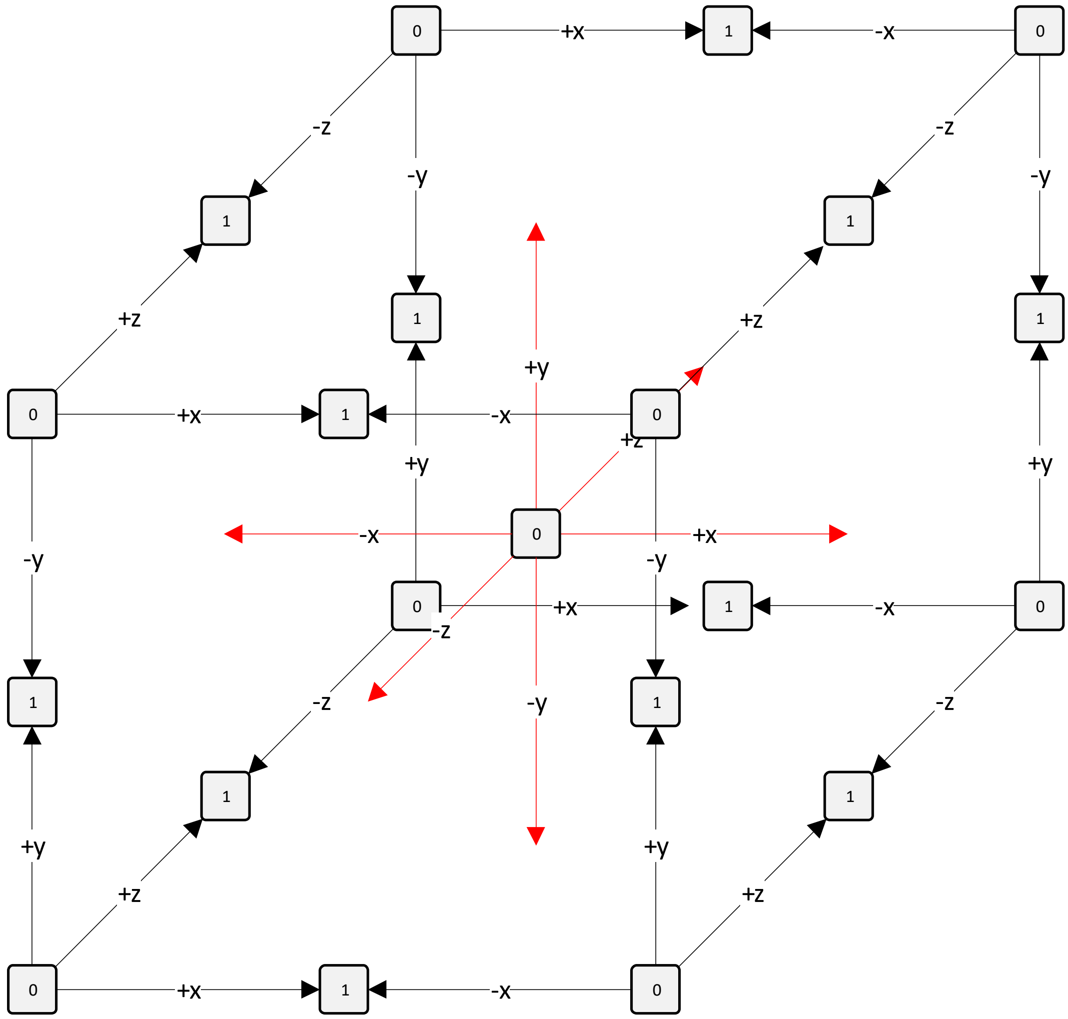

Figure 6.2

Let us look at one quadrant.

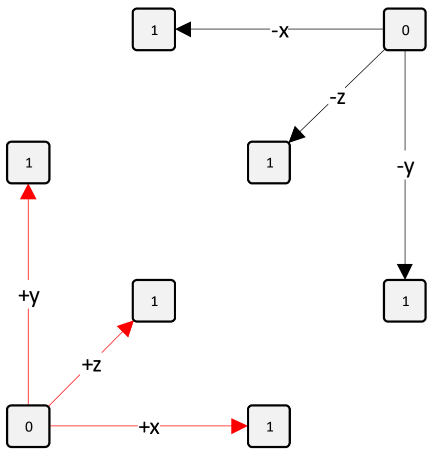

Figure 6.3

A conjecture could be made that a circle is “drawn” by means of bringing a “line” through the following path:

sigma(1,1) -> sigma(4,1)

sigma(4,1) -> sigma(3,1)

sigma(3,1) -> sigma(3,4)

sigma(3,4) -> sigma(3,4) (This would make more sense to consider as sigma(3,4) -> sigma(4,3))

sigma(3,4) -> sigma(3,1)

sigma(3,1) -> sigma(4,1)

sigma(4,1) -> sigma(1,1)

This would result in a circle being “drawn” between sigma(1,1) and sigma(1,1).

Based on the diagram of the structure in Figure 6.2. This would intersect at a point half way between sigma(0,0) and sigma(0,0).

This would result in a shape being “drawn” within the sigma(0,0) -> sigma(0,0) instruction set progression as sigma(1,1) -> sigma(1,1) intersects it exactly halfway through.

Non-Symmetry Resolution

There is a numeric difference here that needs resolving, there are 8 corners to a cube.

There are 2 corners that are sigma(0,0).

There are 4 corners required to "draw" a circle. Since a forward and contra flowing set of instructions are required between 4 sigma(1,1).

There being 4 corners of a cube (sigma(1,1) corners) required to "draw" a shape would suggest that the other 4 corners are sigma(0,0) - symmetry.

Having 6 corners as 1 and 2 corners as 0 would not be symmetrical and not fit the system.

Consider a single quadrant and add the additional trajectories.

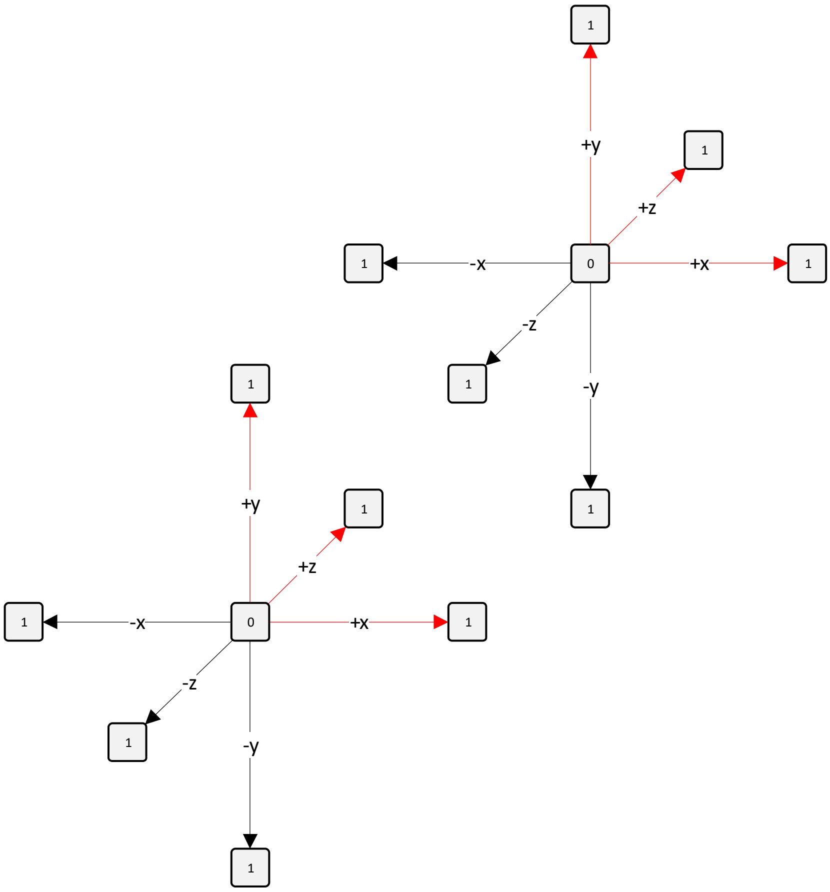

Figure 6.4

If we change any 1 to a 0 or 0 to a 1 to attempt achieving this within the structure created in any diagram before the diagram breaks down.

We effectively need to say: 0 = 1 and we can achieve a structure within which a system that is symmetrical at all levels will work.

This structure being perfectly symmetrical would allow for instruction sets being run in any direction between corner sigma nodes to be intersected by instruction sets being run from other corner sigma nodes(?)

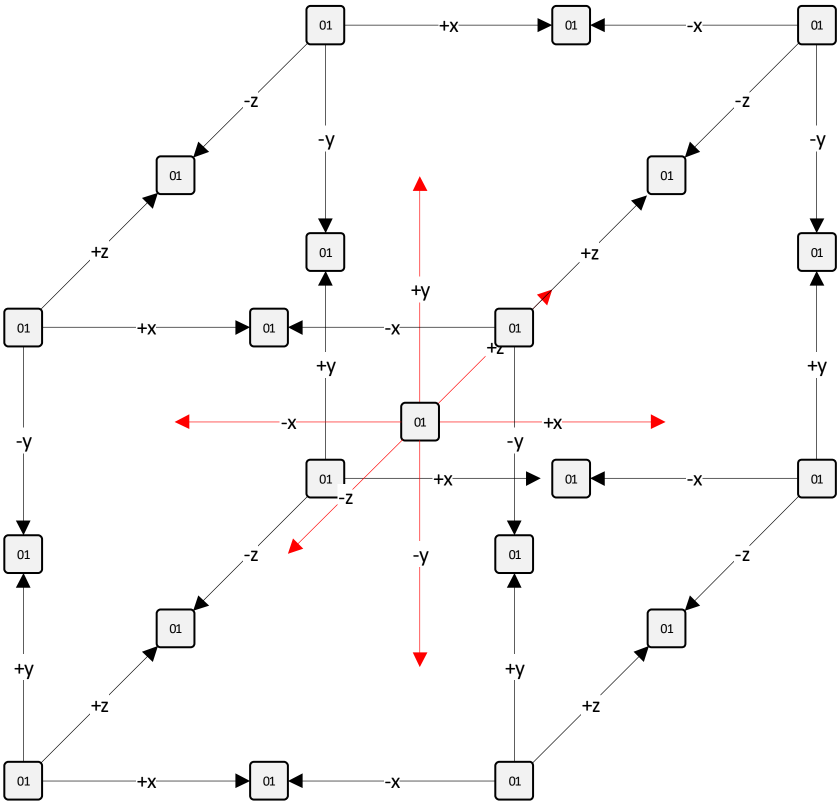

Figure 6.5

Observations

Observation #1

If all points at opposing ends of the hypothetical cubes (quadrant), ie all 8 corners are considered as 0,0,0. The overlapped space would have all instruction sets running within each quadrant in all directions.

You would in this case get contra flowing sets of instructions in every symmetry.

This would result in the ability to combine instruction sets that come from opposing directions as the inverse instruction would be available.

Observation #2

Is a line sigma(X,1) or sigma (1,X)?