Knowing in a World of Broken Symmetries

A Symposium presented at
The Society for Chaos Theory in Psychology and the Life Sciences
17th ANNUAL CONFERENCE
Friday - Sunday, 27-29 July 2007
Chapman University
Orange, California, USA

Part 1
The Puzzling Powers of Two:
A Brief Introduction to Epistemology and to
Boolean Systems Simulation Methodology

Thomas E. Malloy, University of Utah
Jonathan Butner, University of Utah
Joel Cooper, University of Utah
Thomas Smith, University of Utah

*Abstract: Epistemology addresses how we know what we know.   Gregory Bateson's difference-based epistemology is simulated by a Boolean network model.   This formalization has three fundamental premises. First, the map (what humans and other sentient beings know) is not the territory (that which is known); moreover, what gets onto maps from territories are differences in the territories. Second, mental process can be abstracted as a flow of differences in a richly connected network; in Boolean terms, then, James' stream of consciousness becomes a stream of 0's and 1's. This stream of differences in dynamic systems terms can be construed to be an adaptive landscape consisting of many basins of attraction, each with an attractor. Third, higher order knowledge emerges in the process of taking differences in the flow of differences (Bateson, 2000, p. 454ff). The Boolean simulation of these premises produces an emergent landscape of basins of attraction which are construed as the cyclic dynamics of mental process. In Part 1 we present an overview of how E42, our Boolean dynamic systems simulator works and how it maps to epistemology. Particularly, we define the discrete analogues to derivatives based on the XOR operator of symbolic logic which is used to define a matrix function named TAO. TAO generates derivatives expressed as TAO matrices; these TAO matrices are used to analyze the dynamics of the attractor cycles found in the basins of attraction in a Boolean (and by extension, mental) landscape. Like all derivatives, the TAO function can be applied recursively to produce higher order derivatives.   In the last section of Part 1 we report the puzzling result that, when the attractor's period (in iterations) is a power of 2, higher order derivatives diminish to the **0** matrix. When the attractor's period is not a power of 2, higher order derivatives do not resolve in this way. Parts 2, 3, and 4 of this symposium will describe how this result comes in terms of symmetry theory and folding and how the explanation of this puzzling result is key to understanding how (without a homunculus) sentient beings discover new ideas that are within their structural capacity but beyond their previous experience.*

Part 1: Introduction to Simulation Methodology with some Puzzling Results

This symposium will propose one systematic framework for the nature of knowing, that is for epistemology. This framework will be based on the ecological approaches of Bateson and Gibson, on nonlinear dynamic systems models, and on symmetry theory. Our approach is presented in several interrelated parts. In Part 1 we will provide a very brief review of Boolean simulation methodology and analysis for those not familiar with our prior work. The description of our theoretical perspective will be postponed until we present an overview in Part 2; this perspective will be extended and deepened in the following parts of the symposium.

For the moment let us just say that Bateson (2002) proposed that mental process is best described as a flow of differences within a system and within the subsystems that constitute the larger system. As with all flows this flow of differences is patterned. He also proposes that the meta-process of finding differences in the flow of differences will reveal these patterns in the flow of differences. The Batesonian perspective can be modeled by NK Boolean systems (Kauffman, 1993, 1995; Malloy, Jensen, & Song, 2005). We will argue formally from Boolean mathematics that flows of differences are patterned; that is, within a Boolean system, flows of differences define a landscape of basins of attraction. We will review how to take the discrete derivatives of the attractor cycles found in these basins, including not only the first derivative but also higher order derivatives.

We will review some puzzling results not publicly reported before. When we take a series of derivatives, first, second, third,..., eighth,... we find puzzling patterns in these sequences of derivatives. For attractor cycles whose length is a power of 2 the derivatives will diminish to a 0 matrix but for cycles of length not a power of 2 the derivatives will not diminish to 0 but rather the pattern of derivatives (across increasingly higher-order derivatives) will repeat in a cycle forever. The mathematical basis of this result and its epistemological implications will be discussed in later parts of the symposium.

For now we set a simple framework that that we will return to through these papers. Assume there is something like William James' "stream of consciousness" and that like all streams its process forms eddies, standing waves, and the like. In short, assume that the steam of mental process is patterned. Assume also assume that this stream of mental process can be described abstractly as a flow of differences. How can we analyze and describe the patterns within this stream? And if we can find nonlinear dynamical descriptions of this stream, what do they tell us?

Malloy, Jensen and Song (2005) have built an NK Boolean simulation program written in Java and named E42 to study how humans know the characteristics of dynamic systems. To use this software, users must enable Java in their web browsers. Moreover, web browsers on PC's must have the Java plugin. Mac's with OS X and using the Safari web browser have Java is built in; no Plugin is necessary.

NK Boolean networks (described in detail by Kauffman 1995, p. 188ff; Kauffman, 1998, p. 75ff; and Malloy, Jensen, and Song (2005) and extensively at www.psych.utah.edu/dynamic_systems) consist of an arbitrary number N of abstract entities called nodes. Since the model is Boolean, the nodes have two states: ON (state = 1) and OFF (state = 0). Each node takes input (0 or 1) from K nodes. Time flows in discrete iterations. On iteration T, each node, takes the input (0 or 1) it receives from its K input nodes and uses a logical truth table to determine what its own value will be (either 0 or 1) on the next iteration, T+1. That is, the nodes are coupled; the output of one is input to others and visa versa. As a very simple example, suppose a node, call it Node A, receives input from K=2 other nodes. If Node A is using an AND operator then its state will equal 1 on iteration T+1 only if its two inputs are both equal to 1 on iteration T. Similarly if Node A is using an INCLUSIVE OR operator then its state will equal 1 on iteration T+1 if either its first input or its second input or both are equal to 1 on iteration T. So the relationship between a node's inputs at time T determines its state on the next iteration. This simple model can produce complex dynamic patterns whose changes flow across iterations. NK Boolean systems exhibit the usual characteristics of dynamic systems including attractor cycles (basins) which will be described below and which will play an important role in the perception of the dynamics of such systems.

State Vectors and Basins. Malloy, Jensen and Song (2005) describe in detail a small four node (N4) Boolean system. We will skip the formal derivations required for full understanding of that system but we will use some other their results to establish a vocabulary that will be used throughout this symposium. State vectors, **S**(T), are a convenient way to characterize important characteristics of system dynamics. For purposes of exposition, suppose a system (i.e., the one described by Malloy, Jensen & Song, 2005) has but four nodes (N = 4). Suppose on iteration T=1 the ON-OFF pattern for the four nodes from first to last is OFF, OFF, OFF, ON. This can be written as a state vector: **S**(1) ={0001}. The sequence of state vectors **S**(1) = {0001} ✍ **S**(2) = {1100} ✍ **S**(3) = {0011} ✍ **S**(4) = {1000} ✍ **S**(5) = {0001} ✍ **S**(6) = {1100} ✍ **S**(7) = {0011} ✍ etc… therefore describes the flow of states (or, alternatively, the behavior) of the system across time.

Because the system is deterministic a given state vector, **S**(T), will always be followed by the same state vector **S**(T+1). Therefore the fact that in the above example **S**(5) is identical to **S**(1) means that **S**(6) must be identical **S**(2) and **S**(7) must be identical to **S**(3), and so on. This indicates that the system is in an attractor cycle (in this case of length four) because there are four distinct state vectors, **S**(1) through **S**(4), repeating endlessly. This is the attractor cycle for Basin 2 in Figure 1.2, below. The length of an attractor cycle, measured in the number of state vectors before it repeats, is the fundamental frequency of the cycle. We will refer to this as the attractor cycle of Basin 2 in the examples below. The system cannot escape this attractor cycle unless the system is perturbed. (Perturbation amounts to changing the state of one or more nodes.) A fuller discussion of this example is found in Malloy, Jensen, and Song (2005), where it is named 4-Node Standard, or at http://www.psych.utah.edu/stat/dynamic_systems/. The important points here are that the nodes of NK Boolean dynamic systems flow from state to state by a deterministic relational logic, that at any moment the entire system can be characterized by a state vector, and that the flow from state vector to state vector across time can fall into cyclic attractor basins.

Visual Representations of a System's Dynamics

Basin Landscape. We won't derive the structure of the basin landscape here as that is done in Malloy, Jensen, and Song, 2005. But Figure 1.2 shows the results of that analysis and is one way to visualize a Boolean dynamic system. Notice that the dynamics of this small Boolean system fall into three basins of attraction; each basin has an attractor cycle. Basins 1 and 2 have an attractor cycle that repeats every four iterations, we call this a cycle of length L = 4. Basins 1 and 2 also have a few state vectors that are tributaries leading into the attractor. Basin 3 is a cycle of L =1 and has no tributaries. In which basin does a system currently reside? That depends on the initial state vector it starts in and on perturbations to the system. Two different questions that will occupy much of this symposium are the following. Given that a system is in one basin how can it know of the existence of other basins based solely upon information that is local to its current basin? And, if it can know something about the landscape based on its current location, how can it navigate to other basins?

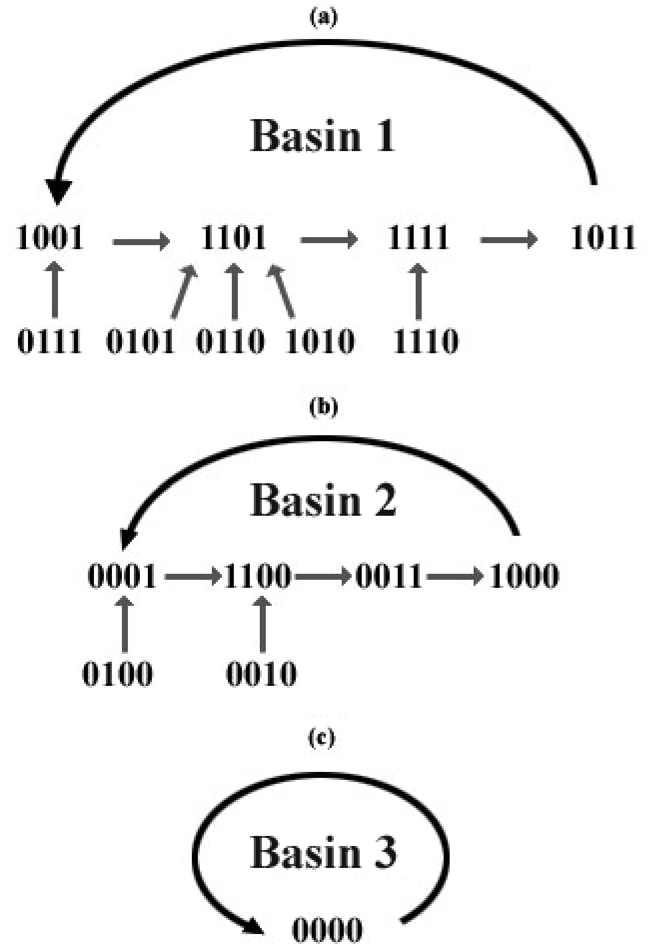


Figure 1.2. The basin landscape of a small 4 node system named 4-Node Standard. The landscape includes three basins, each with an attractor cycle. Some state vectors are tributaries that lead into attractor cycles which repeat endlessly. A basin consists of an attractor cycle and its tributaries.

Historical Trace. There are other ways to represent the dynamics of Boolean systems. Consider the series of state vectors in the preceding discussion and in Figure 1.2. Rotate the state vectors to be column vectors; then put these column vectors on a grid with 0's represented as white cells and 1's represented as black cells. The ordinate will then represent individual nodes from 1 to N and the abscissa will represent iterations from 1 to T. Figure 1.3b shows the state vectors of the attractor cycle for Basin 2 detailed above. The system has 4 nodes (ordinate) whose states vary across sixteen iterations (abscissa). Black cells represent a 1 in a columnar state vector while white cells represent a 0. Notice the visual form generated as the behavior of the system unfolds over time.

Different attractor cycles generate different visual patterns; Figure 1.3a shows the form generated by a second attractor cycle in 4-Node Standard. A fuller discussion can be found in Malloy, Jensen, and Song (2005).

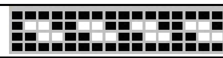
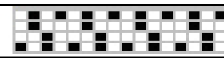
a	**b**
	
Basin 1	**Basin 2**

Figure 1.3. Sixteen iterations (abscissa) of an N=4 node (ordinate) NK Boolean system showing four cycles through the attractor cycles for Basin 1 and for Basin 2. Both attractors are of length L = 4 iterations.

TAO: Discrete Derivatives. Malloy, Jensen and Song simulated the process of differentiating differences by building an analytic tool, TAO, that uses the XOR operator to find the first derivatives of the flow of differences in attractor cycles. Notice that XOR compares two inputs and if the states of the inputs are the same it returns a 0 and if the states of the inputs are different it returns a 1. Therefore we say that the XOR operator detects difference.

Table 1.1. Truth Table for the Exclusive Or (XOR) operator.

Input 1 at T	Input 2 at T	State at T+1
0	**0**	**0**
0	**1**	**1**
1	**0**	**1**
1	**1**	**0**

Consider two state vectors, **S**(1) and **S**(2), from the discussion of Basin 1 above (see text and Figures 2 and 3); these vectors represent the state of the four nodes at T =1 and at T = 2 with the first place in a vector representing the state of Node 1, the second place representing the state of Node 2, and so on. Notice that Node 1 is OFF in **S**(1) but ON in **S**(2); therefore its states are different at T = 1 than it is at T = 2. TAO uses the XOR operator to compare the state of a node across time, returning a 0 if it is the same and 1 if it is different. Therefore the TAO vector in Table 1.2 is 1 for Node 1 (left most value in the vector). The reader merely has to compare the S(1) and S(2) confirming that TAO is 0 when the corresponding values are the same and 1 when the corresponding values are different. Because it tracks the changes in the flow of differences across time TAO is the discrete analogue to the first derivative.

Table 1.2. Two state vectors from the attractor cycle of Basin 2 (see text).	
S(1) =	{0001}
S(2) =	{1100}
TAO-1 =	{1101}

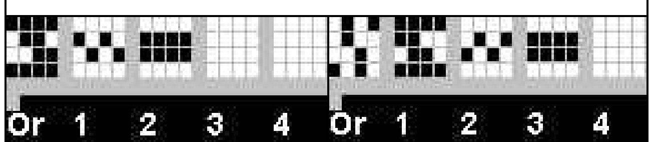
In short, TAO-1 uses XOR to compare each respective position of two state vectors across time. TAO-1 returns a vector of differences in differences (across time). We will consider what we mean by TAO-2, TAO-3, ... next.

Higher order derivatives. We extend the above logic to higher order derivatives by applying the TAO recursively to the output of TAO. Thus we recursively take differences in differences. Table 1.3 shows this process. The reader should be able to confirm the process by inspection.

Table 1.3. Higher order derivatives using Basin 1 as an example.				
Recursive application of TAO				
Flow of Differences		**Differences in {Differences}**	**Differences in {Differences in Differences}**	**Differences in {Differences in Differences in Differences}**
Original flow of State Vectors for Basin 2		**TAO-1**	**TAO-2**	**TAO-3**
T=1	**{1001}**			
T=2	**{1101}**	2 vs1 **{0100}**		
T=3	**{1111}**	3 vs 2 **{0010}**	(3 vs 2) vs (2 vs 1) **{0110}**	
T=4	**{1011}**	4 vs 3 **{0100}**	(4 vs 3) vs (3 vs 2) **{0110}**	[(4 vs 3) vs (3 vs 2)] vs[(3 vs 2) vs (2 vs 1)] **{0000}**
		1 vs 4 **{0010}**	(1 vs 4) vs (4 vs 3) **{0110}**	[(1 vs 4) vs (4 vs 3)] vs [(3 vs 2) vs (2 vs 1) **{0000}**
			(2 vs 1) vs (1 vs 4) **{0110}**	[(2 vs 1) vs (1 vs 4)] vs [(1 vs 4) vs (4 vs 3)] **{0000}**
				[(3 vs 2) vs (2 vs 1)] vs [(2 vs 1) vs (1 vs 4)] **{0000}**

Representing TAO levels visually. The above table can be represented visually. Before we do that we will transpose the row vectors into column vectors. In Table 1.2 and in the above discussion it was convenient to type state vectors as rows due to the nature of printed text so that the nodes run horizontally. But the general convention is to put time (iterations) on the horizontal axis. Transpose the “Original flow of State Vectors” column in Table 1.3 with 1 = BLACK and 0 = WHITE and you will get the “Or” column of Figure 1.4a. Thus, Figure 1.4a shows Table 1.3 with state vectors from the attractor for Basin 1 transposed so that nodes are on the vertical axis and iterations on the horizontal axis. The first column labeled “Or,” for Original flow of State Vectors, shows the four nodes on the horizontal axis, with Node 1 at the top and Node 4 at the bottom, and the four iterations of the attractor cycle for each node along the horizontal axis. Similarly, in Figure 1.4a the “1” column is the transposed representation of the TAO-1 column of Table 1.3, and “3” is the Transposed TAO-3 vectors.   Note that TAO-4 is not show in Table 1.3 because after TAO-3 there is nothing but zeros. Figure 1.4b shows the attractor cycle for the attractor of Basin 2—already shown in Figures 2 and 3—but, of course, this time with its TAO levels visualized. Note that in later analyses we will refer to the original attractor (“Or” in Figure 1.4) as TAO-0, since it is the zero-order derivative.

It is important to note that now we have a qualitative analysis based on visual patterns. Observe that it is relatively easy to see patterns in the flows of difference over time in Figure 1.4. For example, notice the important result that the first derivative of the Basin 2 attractor is the original attractor of the Basin 1. Indeed one deep premise we are exploring is that there is, with certainty in some cases, and possibly in general, a way to discover other attractor cycles of a system by taking the derivatives of the cycle within which the system currently resides. Bateson noted that taking differences in the flow of differences (derivatives) would have epistemological consequences. The epistemological implications of the fact that derivatives—finding the differences in the flow of differences generated by an attractor cycle—can generate other attractors in a system is a tantalizing thought. Thus, Figure 1.4 begins to present a visual description of patterns of differences within the flow of differences.



Basin 1	**Basin 2**
a	**b**

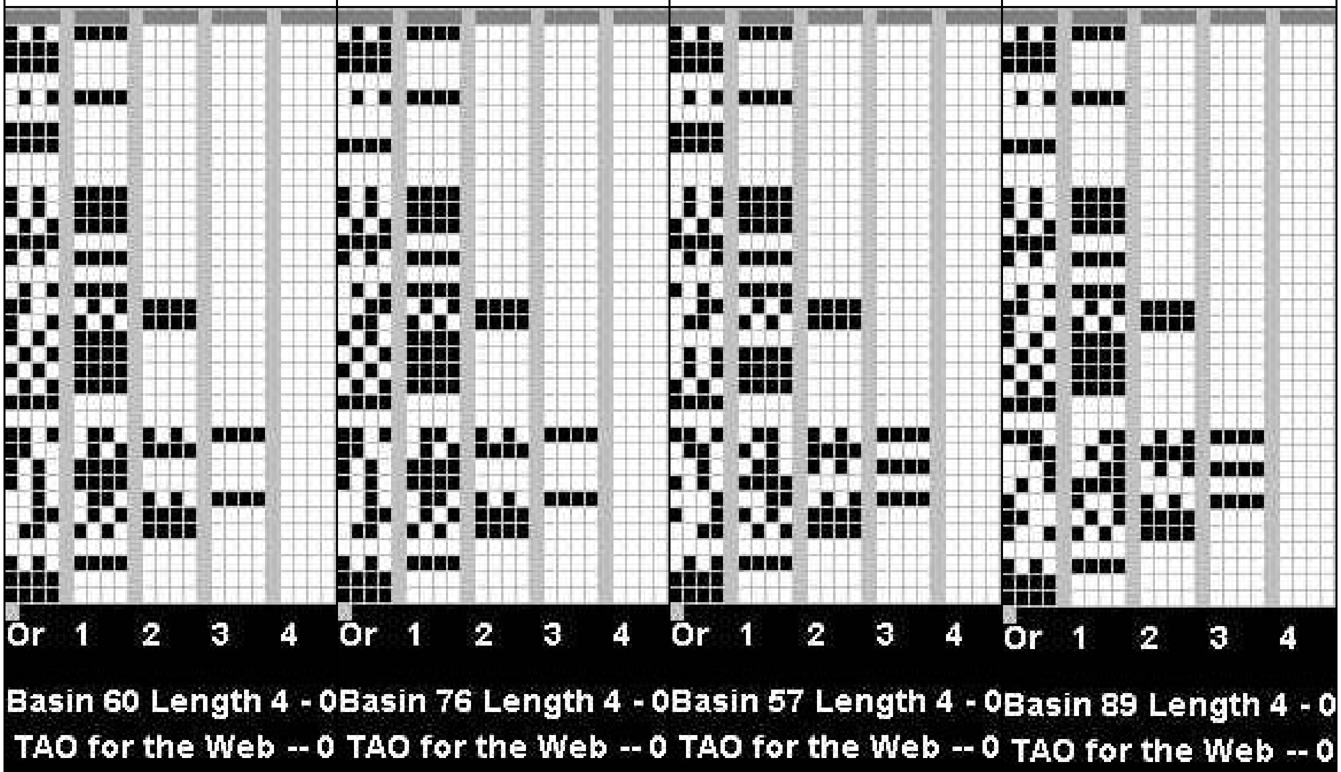
Figure 1.4. The same two attractor cycles visualized in Figure 1.2 but now each cycle is shown with visualizations of its TAO 1 through TAO 4 also. Iterations are on the abscissa and nodes on the ordinate.

Some interesting and puzzling results

Malloy and Jensen (2005) and Malloy, Bostic St Clair and Grinder (2005) have reported the emergence of perceptual hierarchies based on TAO levels; that is, within certain boundary conditions, visual representations of attractors that have identical TAO's appear perceptually similar to humans. These perceptual hierarchies only appear, however, when the attractor cycle length, L, is a power of 2. For L not equal to a power of 2, hierarchies do not appear. In the next paper we will examine the mathematical basis for this distinction between cycle lengths that are a power of 2 and those that are not. For the moment we will review the basic phenomenon.

In Figure 1.5 we see visualizations of four attractor cycles from an N=36 node Boolean system. Figure 1.5 not only shows visualizations of four attractor cycles (labeled #60, #76, #57, and #89) but also the four TAO levels each attractor. It happens that each attractor cycle has a length, L, of 4 iterations, which, of course, is a power of two. As we will establish in the next paper, when L is a power of 2 it is always the case that TAO matrices diminish to **0** and they do so at or before TAO level equals cycle length, L. Certainly that principle is true for the four attractors shown in Figure 1.5.

Attractor of Basin 60	Attractor for Basin 76	Attractor for Basin 57	Attractor for Basin 89



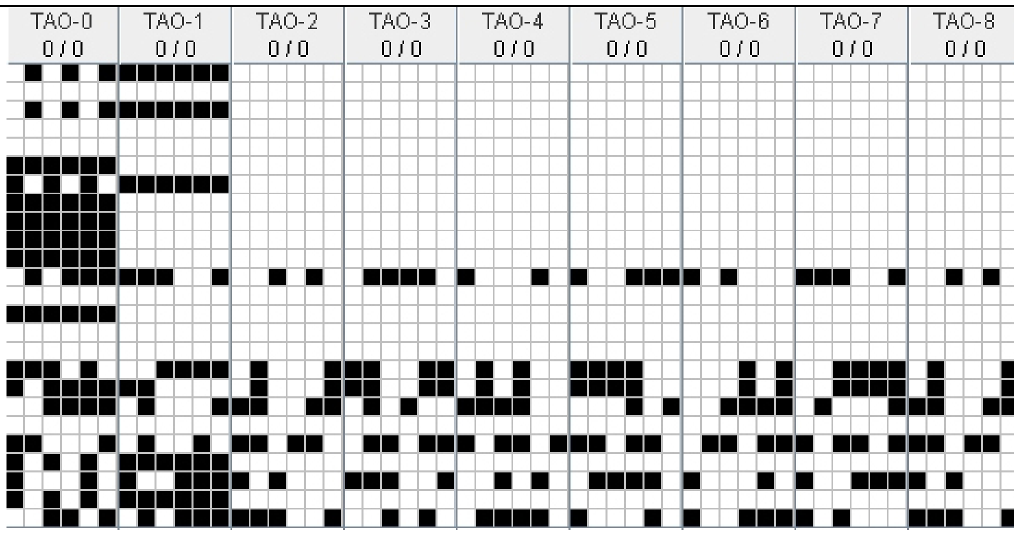
a	b	c	d

Figure 1.5. Four attractor cycles (60, 76, 57, 89) from the landscape generated by a 36 node system Boolean system. The landscape contains more than 100 basins. The four attractors shown have a cycle length L=4 (i.e., L is a power of 2). For each attractor the original attractor (Or) is shown along with TAO-1 through TAO-4. Notice that, like cycles whose length is a power of 2, a higher-order TAO matrix resolves to **0**.

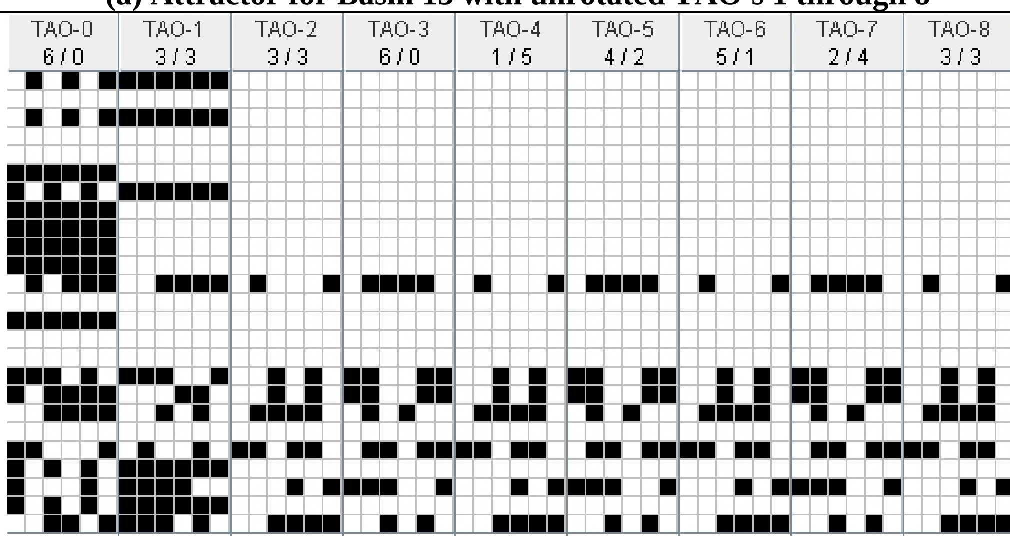
Contrast Figure 1.5 with Figure 1.6. Figure 1.6 shows a single basin's attractor from a different Boolean system; in this case L = 6. The difference between the top and bottom images in Figure 1.6 is that on top (a) the TAO's are presented exactly as calculated while on the bottom the TAO's have have been rotated so each begins with its lowest Boolean value (which makes direct computational or visual comparisons of TAO's much easier). Both the top and bottom image of Figure 1.6 show that the derivative matrices do not diminish to **0** by TAO-6 (the basin length). In fact the derivatives themselves appear to fall into some kind of cycle. This is easiest to see in Figure 1.6 b (bottom). Notice in the bottom image that TAO-2, TAO-4, TAO-6 and TAO-8 are identical as are TAO-3 and TAO-5 and TAO-7. Since the whole system is deterministic, including the calculation of TAO, this alternating pattern will continue indefinitely.

We distinguish between attractor cycle length, L, for the the system taken as a whole and the length of the sub-cycle for individual nodes. Notice that Node 1 (top) for either Figure 1.6a or 6b repeats its WHITE-BLACK pattern every other iteration; in other words for Node 1, its sub-cycle length, sub-L, is 2, which is a power of 2. Therefore the TAO's for Node 1 diminish to **<u>0</u>** at TAO-2. Inspection of Figure 1.6 shows that all nodes that have sub-L equal to a power of 2 follow this rule (we consider sub-L = 1 to be a power of 2). It is those nodes whose sub-L = 6 that produce an overall L = 6 cycle length for all the nodes as a system.

A final observation is that in Figure 1.6 TAO-1 is not part of the repeating pattern of TAO's; it is as if it is akin to a tributary of the cycling TAO's. This example is typical of TAO levels calculated on attractor cycles with L not equal to zero. There are one or two TAO levels that do not repeat and then a series of TAO levels that are repeating. Sometimes there are a large number of TAO levels in the sequence before a TAO level repeats.



(a) Attractor for Basin 13 with unrotated TAO's 1 through 8



(b) Attractor for Basin 13 with rotated TAO's 1 through 8

Figure 1.6. A single attractor cycle for Basin 13 for a small Boolean system with N = 25 nodes. The original attractor cycle is labeled TAO-0 and the figure shows that individual nodes whose sub-cycle is a power of 2 (e.g., top node alternates ON-OFF and thus has a sub-cycle of L=2) resolve to a vector of 0's (all white) with recursive applications of TAO while nodes (e.g., bottom node) do not resolve to **<u>0</u>**.

In Part 2 of the symposium, "Knowing begets Knowing," we take these ideas and methods and propose a new principle of knowledge based on symmetry theory which provides one answer to the following: *A deep question for epistemology is how (without a homunculus) sentient beings discover new ideas that are within their structural capacity but beyond their previous experience.* Subsequent parts of the symposium will examine the fractal basis for the distinction between derivative sequences for powers of 2 and non-powers of 2 cycle lengths and will continue to address how symmetry theory can answer the epistemological question we just posed.