Recursive Distinctioning—Orthodox and Unorthodox

By Louis H. Kauffman

Abstract

This article describes recursive distinctioning (RD) as formulated by Joel Isaacson and articulated by Isaacson and the author. Orthodox RD is discussed and problems and ideas related to it are indicated. Variants of RD are discussed, including the structure of describing describing, formal arithmetic related to Spencer-Brown's Laws of Form and structures of self-reference and recursion related to language, logic, and mathematics.

Keywords: distinction, recursion, recursive distinctioning, form, arithmetic, meaning, syntax, description, self-reference.

I. Introduction

This paper is an introduction to recursive distinctioning (RD).[1] We first give a model for RD. We then discuss other partial RDs and we discuss the role of Spencer-Brown's *Laws of Form*[2] in the articulation of distinctions and recursions. We end with a reflective epilogue.[3]

The theme of this paper is that RD directly instantiates a general dialectic between meaning and syntax, where by syntax I mean any formalism or language that has symbols, signs, or distinctions. It is a circular process, a dialogue. Meaning gives rise to syntax. Syntax gives rise to meaning. In the specific RD actions we describe here, meaning arises in the form of distinctions, and these distinctions become signs, syntax for a next round of distinguishing and meaning, leading again to syntax in an endless round.

This paper is dedicated to the memory of Joel Isaacson and to his deep insight into fundamental process.

II. What is RD?

RD means just what it says. A pattern of distinctions is given in a space based on a graphical structure (such as a line of print or a planar lattice or a given graph). Each node of the graph is occupied by a letter from some arbitrary alphabet. A specialized alphabet is given that can indicate distinctions about neighbors of a given node. The neighbors of a node are all nodes that are connected to the given node by edges in the graph. The letters in the specialized alphabet (call it SA) are used to describe the states

---

[1] Joel Isaacson, Autonomic String-Manipulation System, US Patent 4286330, filed April 26, 1979 and issued August 25, 1981.

[2] George Spencer-Brown, *Laws of Form* (London: G. Allen and Unwin, 1969).

[3] This paper is an extension and modification of Louis H. Kauffman and Joel Isaacson, "Recursive Distinctioning and the Basis of Distinction," *Journal of Space Philosophy* 10, no. 1 (Spring 2021): 69-82.

of the letters in the given graph, and at each stage in the recursion, letters in SA are written at all nodes in the graph, describing its previous state. The recursive structure that results from the iteration of descriptions is called RD. Here is an example. We use a line graph and represent it just as a finite row of letters. The SA is {=, [,], O} where "=" means that the letters to the left and to the right are equal to the letter in the middle. Thus, if we had AAA in the line then the middle A would be replaced by =. The symbol "[" means that the letter to the LEFT is different. Thus, in ABB, the middle letter would be replaced by [. The symbol "]" means that the letter to the RIGHT is different. Finally, the symbol "O" means that the letters both to the left and to the right are different. The SA is a tiny language of elementary letter distinctions. Here is an example of this RD in operation, where we use the proverbial three dots to indicate a long string of letters in the same pattern:

… AAAAAAAAAABAAAAAAAAAA …
is replaced by
… =========]O[========= …
is replaced by
… ========]OOO[======== …
is replaced by
… =======]O[=]O[======= ….

Figure 1. The First Few Steps of RD.

Note that the element]O[appears from the simple difference between B and its neighbors, and that]O[then replicates itself in a kind of mitosis or DNA replication activity.

RD is the study of systems that use symbolic alphabetic language that can describe the neighborhood of a locus (in a network) occupied by a given icon or letter or element of language. An icon representing the distinctions between the original icon and its neighbors is formed and replaces the original icon. This process continues recursively.

Figure 2 illustrates further steps in the recursive process (with a fixed boundary condition). Note the dialectical flavor of the continued patterning. In this model, we have used synchronous processing so that each row is fully worked out before becoming the next row. It is convenient, particularly for pattern investigation, to use synchrony, but it is not necessary. Many asynchronous variations are possible, and we encourage the reader to explore these on his or her own.

```
*AAAAAAAAAAAAAAAAABAAAAAAAAAAAAAAAAA*
*                ]O[                *
*               ]OOO[               *
*              ]O[ ]O[              *
*             ]OOOOOOO[             *
*            ]O[     ]O[            *
*           ]OOO[   ]OOO[           *
*          ]O[ ]O[ ]O[ ]O[          *
*         ]OOOOOOOOOOOOOOO[         *
*        ]O[             ]O[        *
*       ]OOO[           ]OOO[       *
*      ]O[ ]O[         ]O[ ]O[      *
*     ]OOOOOOO[       ]OOOOOOO[     *
*    ]O[     ]O[     ]O[     ]O[    *
*   ]OOO[   ]OOO[   ]OOO[   ]OOO[   *
*  ]O[ ]O[ ]O[ ]O[ ]O[ ]O[ ]O[ ]O[  *
* ]OOOOOOOOOOOOOOOOOOOOOOOOOOOOOOO[ *
*]O[                             ]O[*
```

Figure 2. An extended RD Recursion with Boundary Conditions.

RD processes encompass a very wide class of recursive processes in this context of language, geometry, and logic. These elements are fundamental to cybernetics and cross the boundaries between what is traditionally called first- and second-order cybernetics. This is particularly the case when the observer of the RD system is taken to be a serious aspect of that system. Then the elementary and automatic distinctions within the system are integrated with the higher order discriminations of the observer. The very simplest RD processes have dialectical properties, exhibit counting, and exhibit patterns of self-replication. Thus, one has in the first RD a microcosm of cybernetics and, perhaps, a microcosm of the world.

If we go back to the beginning of the RD and the analogy with DNA, we have a sequence of letters such as

... BBBBBABBBB ...

We then describe them in terms of their mutual likes and differences:

... BBBBABBBB ...
... ===⊃□⊂=== ...
... ==⊃□□□⊂== ...
... =⊃□⊂=⊃□⊂= ...

If the letters on either side of a given letter are equal, I replace the letter by an equals sign (=). If the left side is equal but the right side is different, I replace by ⊃. If the right side is equal, but the left side is different, I replace by ⊂. If both sides are different, I replace by a box (□). Now we can perform recursive distinctioning. Examine the diagram above. We performed the distinction/description process three times, starting with ... BBBBABBBB .... The change from B to A and back to B produced *a protocell* of the form ⊃□⊂, the next description elongated the cell to ⊃□□□⊂, and in the third stage, the protocell divided into two copies of itself! All this comes from making distinctions and describing them with an alphabet so that one can make distinctions again and describe again. Very complex and interesting patterns can arise in this way.

This recursive distinctioning process then reminds us of DNA and how DNA replicates itself. You can think of the DNA molecule as a combination of two strands that we can call W (the Watson strand) and C (the Crick strand). W and C are chemically bonded, and we can denote that by <W|C>. So, we can write symbolically DNA = <W|C>. Special processes enabled by enzymes make it possible for these bonds to be broken and for the cellular environment to supply complementary base pairs to each separate strand. Letting E denote the environment we can write <W|E → <W|C> and E|C> → <W|C>. Thus, we have that in the cell, the DNA molecule can be separated into two strands, each of which then becomes a full copy of the DNA. In symbols this has the pattern:

<W|C> → <W|E|C> → <W|EE|C> → <W|C> E <W|C>.

(Here we allow that the environment can be indicated in place of "nothing" and that it can divide into two parts of environment relevant to the two parts of the helix.)

Compare this symbolic sequence for DNA replication with the Recursive Distinction sequence we just discussed.

⊃□⊂ → ⊃□□□⊂ → ⊃□⊂=⊃□⊂.

The interpretations are different, but the pattern is the same (at an appropriately generous level of observation). This is a place where RD needs research to reveal the deep structure indicated by this commonality of self-replication in RD process and the DNA molecular process.

III. The Concept of RD

We ask the reader to examine the chart in Figure 3, taken from Isaacson's patent document.[4] The chart is a description of the RD process. Note that at a certain point we see described the appearance of a dialectical process, and then the repetition of this dialectic throughout the continued recursion of description begetting description in an endless round. Distinctions are made between surface structure and deep structure and at a certain point in the chart it is indicated that, in the perfect dialectical triad, there occurs the idea of RD. Indeed, the idea of the RD is very much the idea of dialectical process, and in these models, we see the automatic working out of a dialectical process in its most elementary form.

For us, the observers of the simple RD, there is an experience of recognition in seeing that this simple process mirrors the elementary processes of our own thought and discrimination. At that point of recognition, the most fundamental problem arises: What is the source of the distinctions that we perceive?

On the one hand, one can recognize that for a human observer a distinction is always accompanied by an awareness or consciousness of that distinction. Furthermore, it is often the case that what is seen to be distinct depends upon the entire context of the event. A good example is the detection of the blind spot in the eye. This hole in our vision is normally not seen at all, but it can be revealed by looking in a direction to the left of a right thumb with the right eye (left eye closed). Then the thumb can disappear in the visual field, indicating the blind spot, but there is never a hole in the visual field. Some distinctions are distinctions for one modality of perception but not another. All distinctions that humans have are supported by their nervous system, their biology, and the physics of the organism, in addition to the context in which these distinctions are framed. The context almost always involves a language of description, and that language itself is composed of distinctions.

---

[4] Isaacson, Autonomic string-manipulation system.

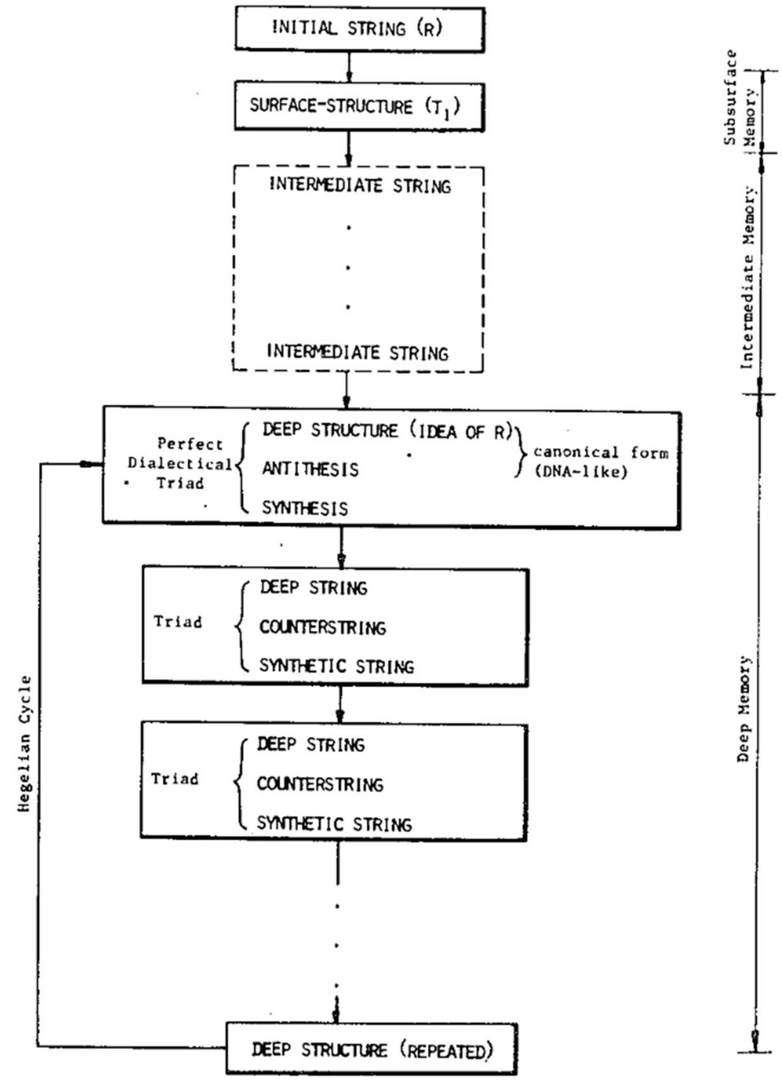


Figure 3. RD Structure and Deep Structure.

Figure 3 can be interpreted as a particularly apt and process-oriented description of the action of distinctions in cognition for a human being. It is also a particularly apt description of the simple RD automaton we described in Section 2. Our intent is not to confuse these two domains but to point out the analogy between them. In a certain sense, the RD automaton engages in a form of cognition and the difference between its cognition and ours is worth contemplating.

By the same token, the RD automaton is based in distinctions that arise in the contiguity of simple elements. In this case, the elements are characters in symbol strings. The analogy can be carried forth to situations in cellular biology where the interactions are those of cells or constituents of cells, and the distinctions have to do with the direct interactions of molecules or with the making and breaking of cellular boundaries. In this arena, significant distinctions are seen to be in operation, apparently independent of our individual cognition and awareness.

This leads to the inevitable discussion of the notion of distinctions independent of human awareness. We understand that such distinctions occur in other organisms and indeed within our own organism. The digestive system also makes its distinctions in regard to the food we hand it, and thereby enables the continuance of the body. My computer also does its operations, independent of my possible understanding of its programming.

The RD automaton can suggest, in this field of analogies, that certain processes of distinction and indeed language precede the consciousness that we take to be the locus of distinctions for our understandings. Some reflection may convince the person who thinks about these ideas that the conception of distinction is circular. Distinctions beget distinctions in an endless round. And once again the RD automaton is a simple model of that dialectical process.

IV. Synchrony

RD processes, as we have discussed them, are synchronous processes in the sense that several variables (the characters in a string for example) are replaced at the same time by a globally defined rule. It is also possible to discuss and investigate asynchronous processes where the updating occurs locally and in different orders than the simultaneity we have imposed. Nevertheless, in this discussion, we adhere to synchronous processes and leave the asynchronous for another time (*sic*).

The general synchronous process is described very succinctly in mathematical terms. Let there be given a set of variables $x_1, x_2, \ldots, x_n$ and a collection of functions $F_1, F_2, \ldots, F_n$ where each F_k is a function of these variables. We may write $F_k(x_1, x_2, \ldots, x_n)$. Then we define a synchronous process where the variables are updated by the equations

$$x_1' = F_1(x_1, x_2, \ldots, x_n)$$
$$\ldots$$
$$x_n' = F_n(x_1, x_2, \ldots, x_n)$$

If we let x denote the vector of values $x = (x_1, x_2, \ldots, x_n)$, then the system can be written concisely as $x' = F(x)$ where F is the vector of F-values. The crux of a synchronous process depends on the choice of the rules of $F(x)$. In the case of the RD process described in Section 2, we have an F that is defined on triples of values (a character and its neighbors to the left and to the right). The possible values are the symbols], [, =, and O and any other distinguishable character symbol. We have:

F(A, B, C) = O
F(A, A, C) = [
F(A, B, B) =]
F(A, A, A) = =

Where A, B, and C denote signs so that A, B, and C are distinct.

The output of the function of three variables is the new character that replaces the middle variable. In a long string of characters, this computation is performed for each triple and the results are stored until all computations are complete. Then the new row of characters replaces the original row. This is the synchronous model for the one-dimensional RD.

Note that the new characters (O, =, [,]) are iconic for the distinctions they represent. Thus, this orthodox RD has the characteristics of distinctions involving adjacency and iconicity. Each new character is an icon for the distinction that it connotes. These properties single out this RD from the vast collection of possible recursions, even those that involve only four values and three variables.

It is interesting to examine the simplest examples of the RD recursion. For example, we can use strings of length three with the boundary condition that the end characters are always seen to be different from the emptiness on their right or their left. Then we have a period two oscillation as shown below.

ABC
OOO
[=]
OOO
[=]
OOO
...

If we use strings of length four, then we can attain a period three oscillation.

A B C D
OOOO
[= =]
O [ ] O
OOOO
[= =]
O [ ] O
...

There is a range of periods and behavior to be explored in this simplest RD.

If we consider functions at this same level of simplicity, then some analogous behaviour can be observed. For example, let 0 and 1 denote the two basic Boolean values and let $<x>$ denote the negation of x so that $<0> = 1$ and $<1> = 0$. Then we can define a simplest recursion by $x' = <x>$, leading to a period two oscillation … 01010101 … A next simplest example that leads to a period four oscillation is

$$x' = y,$$
$$y' = <x>.$$

Beginning here we can construct many oscillators and many patterns. The RD phenomenon occurs very near the beginning of this hierarchy of mathematical possibilities.

In fact, all the processes of the form $x' = F(x)$ can be seen as RD. It is a matter of investigation of the details of the recursion $F(x)$ to find out how the rules of these distinctions operate. A good arena for examining this is the field of cellular automata, where experimentation with rules has led to a vast zoo of phenomena. Not all such recursions take part in the dialectical process of the RD, but all are available to be seen as the consequence of making distinctions and expressing them in a recursive domain.

V. The Audioactive Recursion

1
11
21
1211
111221
312211
13112221
1113213211
…

Illustrated here is a pattern of *recursive description*. Each line is a description of the previous line. To see this, read the lines aloud. The second line says, "one one," and that is a description of the first line. The third line says, "two ones," and that is a description of the second line. The next line says, "one two, one one," then "one one, one two, two ones," and so on. The full alphabet for this recursion is the set of numerals {1, 2, 3}, and these are alternately signs and elements of the description of a pattern. This "audioactive sequence" was extensively investigated by John Horton Conway,[5] and it has many mathematical properties.

[5] John H. Conway, "The Weird and Wonderful Chemistry of Audioactive Decay," *Eureka* 46 (Jan. 1986): 5-16.

A variant on the above recursion that is quite interesting starts with the number three rather than one. Then we have

3
13
1113
3113
132113
1113122113
311311222113
…

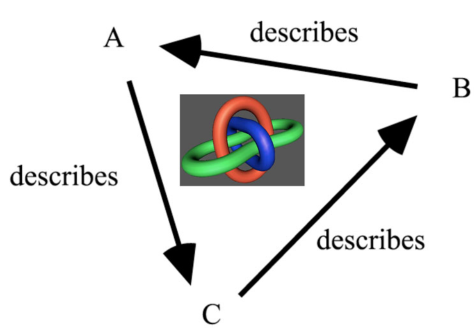
It is not hard to see that if the rows are r_1, r_2, r_3, … then r_{n+3} is an extension of r_n. This means that we can build three infinite rows A, B, C that are in dialogue with each other in the sense that B describes A, C describes B, and A describes C.

A = 111312211312 …
B = 311311222113 …
C = 132113213221 …

There is much to explore in this recursion. A description is of course certainly a distinction, but the distinctions made by this form of description are of a more complex nature than the adjacencies in the first RD that we have discussed.

Remarkably, the audioactive sequences shown here are based on a very small alphabet of numerals (1, 2, 3). It is a bit mysterious what can come from only one, two, and three.

A = 11131221131211132221 …
B = 3113112221131112311332 …
C = 132113213221133112132123 …



We have two mappings defined on strings of digits that take strings of digits to strings of digits. For symbolic sake, let *S* denote the collection of all finite strings of digits. *D*:*S* → *S* is our "descriptor" and *U*:*S*′ → *S* is our "undescriptor." U is only defined on those strings *S*′ that are descriptions.

Examine 22. Its description is 22. Its un-description is 22. It is a perfect self-describer. *D*(22) = 22. *U*(22) = 22. The description of two twos is two twos. We can compare how 22 produces itself with John von Neuman's machine *B* that can build itself! The universal von Neumann machine *B* is a "universal builder". Give *B* a description *x*, and *B* will build the entity *X* with that description. So, one would write

$$B, x \rightarrow X, x.$$

B would use the blueprint *x* to build *X* and produce *X* along with its blueprint *x*. This is fantastic. *B* can build itself. You just give *B* its own blueprint, *b*! Then *B*, *b* → *B*, *b* and *B* produces a copy of itself.

$$B, b \rightarrow B, b.$$

Let us take the arrow *nx* → *xxx* … *x* (*n* *x*es) to mean the "un-describe" arrow that produces the string whose description is *nx*. This is the analog of what a building machine does, and *nx* is the blueprint. Then we have 2*x* → *xx* and we see that 22 → 22 builds a copy of itself. This is of course a special case of von Neumann's pattern. There is also a 2 in the von Neuman machine. He has *B*, *x* → *X*, *x*. Two entities produce two entities. So, *B*, *b* is really a repetition, just like 22, where the two twos in 22 are different. One says the number of twos in the entity that is being described.

VI. Formal Arithmetic

Here we give an example of *formal arithmetic,* governed by a very simple recursive distinctioning with contiguity of characters. The formal arithmetic rules for changing a string of characters consisting in the characters "*," "<," and ">" are as follows:

** is replaced by <*>
>< is replaced by (nothing).

Note that in this recursion, we rely on adjacency to detect the patterns that are to be replaced. Detection and replacement of pattern is the form of distinction in this model.[6]

If we start with a row of five stars, then the following recursion will occur.

*****
<*><*>*
<**>*
<<*>>*

[6] Louis H. Kauffman, "Arithmetic in the Form," *Cybernetics and Systems* 26, no. 1 (1995): 1-57.

If you interpret * as the number 1, <*X*> as 2*X* for any number *X*, and *XY* (adjacent strings) as *X* + *Y*, then you will see that result of the string replacement will be a coding of the number of stars in the first row. In this example, <<*>>* = 2(2(1)) + 1 = 4 + 1 = 5. In fact, the result of the recursion can be interpreted as the binary coding for the original number of stars. Here is another example:

****
<*><*><*><*><*><*><*><*>*
<********>*
<<*><*><*><*>>*
<<****>>*
<<<*><*>>>*
<<<**>>>*
<<<<*>>>>*

The result tells us that there are $2^4 + 1 = 17$ stars in the first row.

Here is the method to convert the result of the recursion to binary notation. Start with the result. Remove the left pointing arrows. Replace the stars by instances of 1. Place a 0 in between each >> and place a 0 at the right if there is no star. Then remove all the right pointing arrows.

For example:

<<<<*>>>>*
>>>> 1>>>>1
1>0>0>0>1
10001.

This recursion is a simple automaton that does arithmetic and converts numbers into binary. Everything proceeds from two forms of distinction. One form recognizes pairs of stars and replaces them by a bracketed star. The other recognizes oppositely pointing pairs of brackets and erases them. At first, it is not obvious that these two forms of distinction are a basis for calculations in arithmetic. Just so, there are recursive processes behind our familiar actions that would seem unfamiliar until we examine them. Consider an everyday action such as speech and ask yourself how you produce the highly patterned sounds that constitute your voice. It is a long story in new territory to articulate what happens in that domain.

Recursion in arithmetic is itself unknown territory for most mathematicians and scientists at this time. For example, consider the following Collatz Rule:

If *n* is even, replace *n* by *n*/2.
If *n* is odd replace *n* by (3*n* + 1)/2.

If n = 1, STOP.

For example, 7 > 11 > 17 > 26 > 13 > 20 > 10 > 5 > 8 > 4 > 2 > 1. It is conjectured that for any natural number n, this process will, after a finite number of steps, terminate at 1. The problem has been known since the 1940s. It remains unsolved at the time of writing. Many adventures can be had in exploring the Collatz Recursion. It is based on little more than elementary arithmetic and the distinction between even and odd. I have used the arithmetic automaton based on a star and bracket to explore the Collatz problem, but it has not yielded up its secrets yet. This problem indicates the depth of simple recursions in the structure of elementary mathematics. Mathematics itself is built from distinctions. We are often surprised by the phenomena that emerge just from mathematics itself in the face of recursion.

VII. Laws of Form

This example is different than the previous ones. Here we start with distinction, but we do not institute rules for a synchronous recursion. The system we describe is due to G. Spencer-Brown in his book *Laws of Form*.[7]

Here the sign ⏋ stands for the distinction that it makes between inside the sign and its outside. Spencer-Brown calls ⏋ the mark, and allows it to refer to any given distinction, including itself. The inside of the mark is unmarked. The outside of the mark is marked (by the mark).

The mark a⏋ can be interpreted as an instruction to cross the boundary of a distinction. In that mode, we have denoted the value obtained by crossing from the state a. Thus ⏋⏋ is unmarked, since we have crossed from the marked state, and ⏋ is marked since we have crossed from the unmarked state. An extra mark in the space outside the mark is redundant since that space is already marked. Consequently, we may write ⏋⏋ = ⏋. Thus, we have two basic replacement rules:

Crossing: ⏋⏋ =

Calling: ⏋⏋ = ⏋.

A calculus arises from this so that one can reduce or expand arbitrary expressions in the mark. For example,

One can prove that the simplification of an expression is unique and go on to consider the algebra that is related to this arithmetic.

---

[7] Spencer-Brown, *Laws of Form*.

In the algebra, we have identities such as AA = A for any expression a, and $\overline{\overline{A}\rceil}\rceil$ = A for an expression A. Remarkably, the algebra is quite non-trivial and leads to a new construction for Boolean algebra and new insights into the nature of logic.

Here, a great deal of structure comes to light if we decide not to use synchronicity immediately and to elicit designs that are asynchronous and have behaviour that is independent of choices of time delay. In this way, the timeless structure of such asynchronous structures enters and supports the creation of the rhythm and temporality of recursive computation. In this way, one can consider recursive structures related to the calculus of indications (as this calculus of the mark is called).

An elementary structure of great significance appears from these equations:

$$M = \overline{aN}\rceil$$
$$N = \overline{bM}\rceil$$

To see what happens here, let *a* and *b* be unmarked. Then we have

$$M = \overline{N}\rceil$$
$$N = \overline{M}\rceil$$

If M = $\overline{}\rceil$ and N = $\overline{\overline{}\rceil}\rceil$, these values satisfy the equations, and so the system is in a stable state. Similarly, if M = $\overline{\overline{}\rceil}\rceil$ and N = $\overline{}\rceil$, then the system is in a stable state. We see from this that M and N together form a memory. In a possible world of recursions, the memory can maintain a particular pair of values. In this way, the binding of structure across time emerges from the timeless eternity of forms.

Furthermore, if in this memory we were to change *a* or *b* to the marked state, we could influence the memory to change state. A momentary change in the inputs *a* and *b* can reset the memory. In this way, circular systems of equations can be made that correspond to circuitry at the base of computing, and the essential design of digital computers can be accomplished in the language of the mark and its algebra.

A key function that can be described in this algebra is the operation of exclusive or. We denote exclusive or of *A* and *B* by *A* # *B*. It is expressed in the algebra of the mark as:

$$A \# B = \overline{A\overline{B}\rceil}\rceil\overline{\overline{A}\rceil B}\rceil.$$

The reader will note that *A* # *B* is marked only when one of *A* or *B* is marked, but not both. Thus *A* # *B* can indicate whether *A* and *B* are distinct or not. If *A* = *B* then *A* # *B* is unmarked, but if *A* ≠ *B*, then A # B is marked. It is this ability of the logical algebra to make distinctions that gives it the capacity to be the underpinning for models of recursive distinguishing.

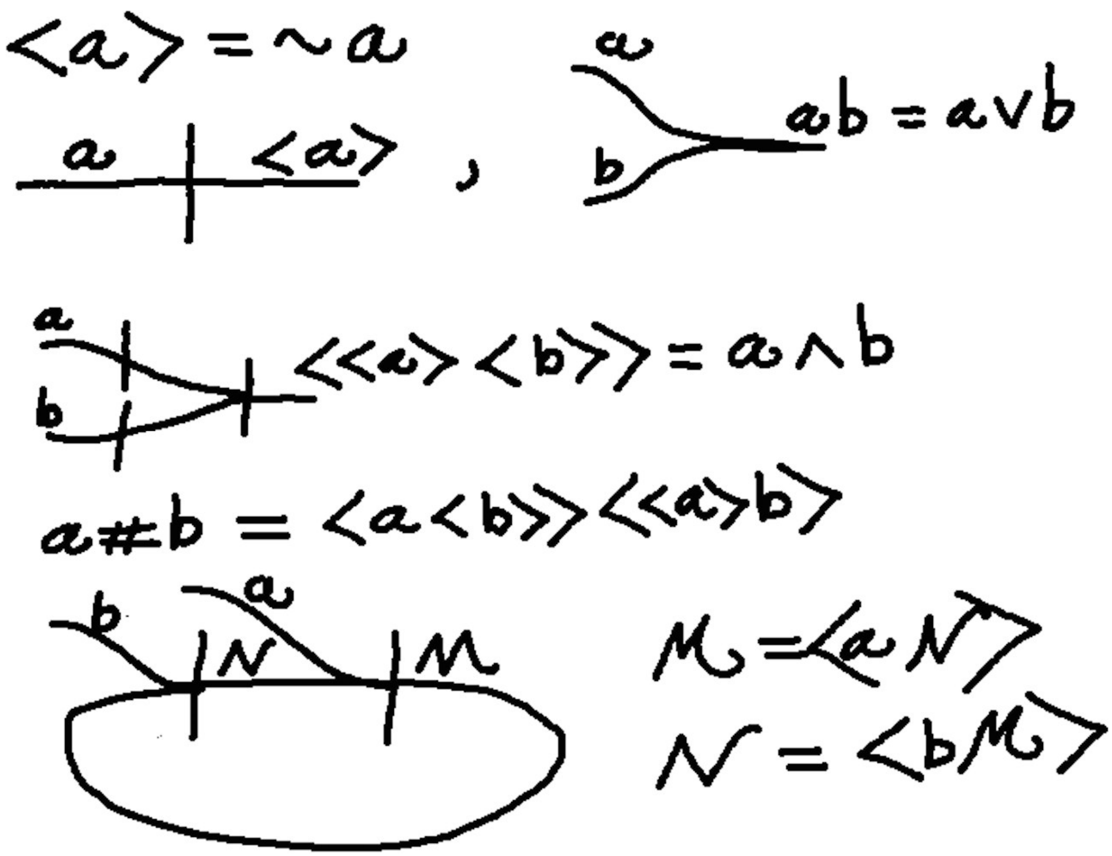


Figure 4: Diagrams for Distinction Operators

In Figure 4, (using <> for the mark) we indicate the bare bones of diagrams for these distinction operators and for the memory. It should be apparent that the memory can be regarded as a graph with a special even cycle that can be labeled with states so that there is an evaluation balance at each node. The node with the vertical marker is the distinction operator, and it is an analog of a NOR gate in electronics. Systems composed of such diagrams can be used to model the basic workings of any digital computer, and so make a Turing complete structure. In this way, we see that all computation can be seen to be based on distinctions and recursions. This way of creating a basis is not quite in the mold of RD where all distinctions are created in relation to contiguities and the formation of alphabets. In this circuit paradigm the distinctions act as states and transmissions of elementary information in the cycles and trees of graphical structures that are themselves seen as patterns of distinction operators.

By regarding the distinction operators as graphical carriers of information, the structure of these graphs in Laws of Form can be shaped as models of automata that can be built in hierarchical fashion and so concatenate into full blown designs for digital computers and information systems. By the same token, these designs can support the operations of any RD of the type that we have described in this paper. We can enfold the RD concept and designs into a full context of computation and communication.

In this way, we come full circle for the structure of RD in that the consideration of a distinction and the evolution of an algebra and operations of distinction creates the platform on which RD can be constructed. But the process by which we have evolved this algebra and logic is, in fact, the already given RD capabilities of our organism and our abilities to make engineering and mathematical design.

We can reach deeper into the biological and physical world to find sources that underpin the emergence of distinctions. This will inevitably happen in the future development of RD and the understanding of distinction.

VIII. The Intellector

The intellector in the Isaacson Patent[8] is based on the operation of an XOR on a pair of entities. We discussed this operation in Section 7 of this paper. Writing *a#b* for exclusive or, we have that *a#b* is marked if *a* is distinct from *b*, and *a#b* is unmarked when *a* is equal to *b*. The key point about *a#b* is that it is given as a sensitivity to same or different in a possible situation where a direct observation of *a* or *b* would not suffice to give that information. Thus, *a#b* is relative information. The intellector is at the base of the construction of the RD, and it can detect difference and so begin the process of recursion. The intellector does not have to detect the values of individual entities, only whether they are the same or different for its information gathering capacity. This property of starting with relative information is very important for both the epistemology of RD and its possible applications. After all, a bacterium cannot name the components of its environment, but it can interact with them.

Note that the intellector processes *streaks*, rather than strings written in the symbolic alphabet (the four icons). We illustrated this in Section 2 with input strings of the form BBBBABBBB. Such strings can contain any characters whatever just so long as the intellector can discriminate identity or difference between any two of them. A streak codes for same/not same between adjacent entities in strings. Thus, a streak may represent any string whatsoever, including strings of fantomarks (not directly detectable marks). Thus, things are removed from directly observable signals to binary streaks that represent difference and sameness. All this is described in the patent.[9]

In the patent, the term intellector denotes an electronic circuit that is built of multiple XOR gates. Further distinctions about coding and processing of strings occur in the patent, to which we refer the reader. Note that in a natural RD process, we do not expect electronic circuits or algorithmic string manipulations to model the intellector, but there will be relative ways for discrimination to arise and produce new entities to be discriminated.

---

[8] Isaacson, Autonomic String-Manipulation System.

[9] Isaacson, Autonomic String-Manipulation System.

IX. Language, Reference, and Self-Reference

In an earlier section, we discussed a simple construction of self-reference in which "two twos" describes "two twos". This is a non-paradoxical self-reference of ordinary language to itself. The discussion also re-described this aspect of describing describing. We can take this discussion again through a simple language in which this pattern occurs. The words in this language are all the strings S, SS, SSS, SSSS, … ad infinitum.

A string of the form *SX* is given to refer to the string *XX*. Here *X* denotes one of the strings above. Since *SX* refers to *XX*, we see that *SS* refers to *SS*. This is the essence of the matter. The feedback loop completes from *SS* to itself, just as from 22 to 22. From here you see that the Russell paradox builds on this pattern with the Russell set defined by the equation $Rx = \sim xx$ so that substituting *R* gives the self-denial $RR = \sim RR$. By the time Church and Curry had abstracted the essence of the Russell paradox, they had taken the view that RR is an entity that is invariant under negation. *RR*, being paradoxical, is in the language of cybernetics, an eigenform for negation, an imaginary logical value, a token for the process of negation. The Russell Paradox eigenform *RR* became a valued member of non-standard mathematical discourse.

In examining describing describing, we indicated a route to self-reference and eigenform There is another route, fundamental to language and communication. I call this process the *indicative shift*.[10] Let

$$b \to B$$

denote a *reference of b to B*. You can take *b* to be a name for *B*. The *shift* of this reference is denoted by

$$\#b \to Bb.$$

I am introduced to *B* and my host says to me, please meet "*b*." Being an attentive guest, I make an association of the name *b* with the appearance *B* and put them together in my mind. The next time I meet *B*, he appears to me as a *Bb* in the sense that the name comes right along with him. In my imagination, *B* has the name-tag *b* on his lapel. I also have that name stored with a marker *#b* (made explicit here but usually unsaid). The name *#b* is now pointing to my amalgam of *B* with his name. To examine your own process in this regard, think of the times you have encountered someone whose name you have forgotten. You find yourself attempting to reconstruct the links we have just described.

It is possible for a sign to stand for itself. For example, a bracket < > can be regarded as a sign for the distinction that it makes between its inside and its outside. In that case

[10] Louis H. Kauffman, "Self-Reference and Recursive Forms," *Journal of Social and Biological Structures* 10 (1987): 53-72; Louis H. Kauffman, "Eigenform," *Kybernetes* 34 (2005): 129-50; Louis H. Kauffman, "Categorical Pairs and the Indicative Shift," *Applied Mathematics and Computation* 218 (2012): 7989-8004.

< > is a sign for itself. We can write <>→ <> to indicate that the bracket refers to itself. Shifting, we have

#<> → <> <>.

In G. Spenser-Brown's work, *Laws of Form*,[11] he takes the self-referential mark of distinction as a starting point and the equations

<><> = <>
<<>> =

as the expression of a calculus of indications. The first equation can be interpreted as the redundancy of calling the name of the mark by itself. If I wear a name tag, it is not necessary. The second equation involves regarding the sign <> as an act of distinction or crossing. To cross from the unmarked state achieves the marked state: <> = <>. To cross from the marked state achieves the unmarked state <<>> = . For a minimal formalism in the enactment of meaning and syntax, one could not ask for less than the calculus of indications of George Spencer-Brown. Contraction of reference leads to expansion of awareness.

What does the indicative shift have to do with self-reference? You even have a name for the shifting process itself. So, suppose that *M* is the name of the shift. Then

M → #.

Shifting, we find that

#*M* → #*M*.

The meta-name (#*M*) of the name of the shift refers to itself.

We rewrite:

I am the meta-name of my meta-naming process.

and find a relative of the Heinz von Foerster sentence

"I am the observed relation between myself and observing myself."

You can begin the indicative shift at the most elemental point.

→

[11] Spencer-Brown, *Laws of Form*.

This is an arrow from nothing to nothing. Nothing stands for nothing. An arrow prior to names. Pointing without content. It would be self-referential if there were a self in nothing.

Apply the shift.

$$\# \rightarrow$$

There was nothing to shift.

Apply the shift again

$$\#\# \rightarrow \#$$

and again

$$\#\#\# \rightarrow \#\#\#.$$

Thus, we have

$$\begin{array}{c} \rightarrow \\ \# \rightarrow \\ \#\# \rightarrow \# \\ \#\#\# \rightarrow \#\#\#. \end{array}$$

At the third departure from the void, we find that self-reference has occurred.

Meaning arises from syntax in our understanding of the process that it connotes.
In the beginning there was no name.
The shift became the name of nothing.
The shift of the name of nothing became the name of the shift.
The shift of the name of the shift is its own name.

Here begins a cybernetics of the acts of distinction.

The self-reference of the indicative shift is more subtle than two twos. It involves the action of observing and it shows how the act of observing (and naming) turns around and names itself. We are aware of the realm of meaning for the observer and, here we have begun steps into a syntax for the observer

X. The Gödelian Shift

The indicative shift occurs in reference and naming in language and communication. A shift of this shift takes us to the key structures of Gödel's Incompleteness Theorem.[12]

---

[12] Kauffman, "Self-Reference and Recursive Forms"; Kauffman, "Eigenform"; Kauffman, "Categorical Pairs and the Indicative Shift."

Kurt Gödel proved that no consistent formal system rich enough to handle number theory could be complete. Gödel showed that there are true theorems about numbers that the given formal system cannot prove.

Gödel produced a sentence that encoded a denial of its own provability. He devised a method to code each formula F in his system with a number $g = g(F)$ (the Gödel number) so that the formula could be uniquely decoded from its number. I write $g \rightarrow F$ to denote that "g is the Gödel number of F."

Now suppose that $F(u)$ is a formula with a free variable u. For example, $F(u)$ could be "u is a prime number." Let g be the Gödel number of $F(u)$. Then we can substitute g into $F(u)$ to obtain $F(g)$. This is a new formula with a new Gödel number, call it $\#g$. Then we have $\#g \rightarrow F(g)$. This is the "Gödelian indicative shift" of $g \rightarrow F(u)$.

$$g \rightarrow F(u).$$
$$\#g \rightarrow F(g).$$

Now the function assigning the number $\#g$ to the number g is an algorithm about numbers, just the sort of thing that Gödel's formal system L can talk about. Thus, we can have # as an element in the language of L.

Let $B(u)$ be a statement in L that asserts the provability of the statement with Gödel number u. Then $\sim B(u)$ *asserts the unprovability of the statement with Gödel number u*. Furthermore, we have a Gödel number for $\sim B(\#u)$, the statement that the formula with Gödel number $\#u$ is not a provable formula:

$$g \rightarrow \sim B(\#u).$$

Then making the shift, we have

$$\#g \rightarrow \sim B(\#g).$$

This shows that $\sim B(\#g)$ asserts the unprovability of the formula with Gödel number $\#g$. But that formula is $\sim B(\#g)$! This means that $\sim B(\#g)$ asserts its own unprovability. If L could prove this formula, then L would be inconsistent. We assume that L is consistent and conclude that it cannot make the proof. But that is exactly what the formula says, and so $\sim B(\#g)$ is true but not provable in L.

We have sketched the proof of Gödel's incompleteness theorem and, as you see, the Gödelian indicative shift is the key mechanism whereby self-reference is achieved to obtain a theorem that asserts its own unprovability. The self-reference is accomplished via the coding of texts to Gödel numbers and so is protected from paradox. Here the pendulum swings wide from the meaningful arena of the indicative shift in the naming processes of ordinary language to the highly syntactical regions of formal systems. I say that when we are willing to engage such wide swings and are willing to attend to both the large meanings and the formal detail, then the scope of cybernetics and second-order cybernetics is really taken up and challenged.

The Gödel Theorem deserves to be seen as a result of observing systems, systems that embody both formalism and understanding. Its content requires an observer and his or her understanding of the formalism. The Gödel Theorem requires the rational comprehension of an observer. In working with the final shifted equation

$$\#g \rightarrow \sim B(\#g)$$

we stand outside the formal system, understanding the meaning of the reference that makes $\sim B(\#g)$ state its own unprovability. We prove that *L* will produce a contradiction within its own syntax if *L* should produce a demonstration of $\sim B(\#g)$. We reason structurally about *L* through our relationship with *L*. We have access to the properties of the shift and an ability to reason about it that is not available to *L*.

We begin to understand how we as observers can be in intimate relation with a formal system. We can go forward into the classical and deep questions of the relationship of ourselves and machines (formalities, syntax). We return as always to the feedback loop of meaning and syntax and see this relationship anew.

XI. Epilogue

We should say a bit more about dealing with the line between what can be formalized, what is not (yet) formalized, and perhaps what cannot be formalized. Distinction cannot be formalized. This is because a definition is a special form of a distinction. So, any definition of distinction would be limiting the concept. This in no way inhibits us from pursuing distinctions. We must understand that any given formalization is not everything. No model fully encompasses what it would purport to describe. No artifice will capture nature. No artificial intelligence will capture intelligence. And yet intelligent behavior can arise in the simplicity of recursive distinction. Recursion arises when distinctions interact to produce new distinctions. Distinctions arise and distinctions interact to produce distinctions. Processes of this sort are at the base of all structure and the evolution of structure.

What are the fundamental distinctions? Where do they come from?

The orthodoxy of a specific formalism is just that, an orthodoxy. We will find out more as we keep looking and feeling and theorizing and inventing and discovering. Distinctions, both unaware and aware, arise. We pointed out that contiguous elements in strings or other grids may give rise to distinctions. In nature, the act may have little to do with formalism. Recursion in systems of distinctions tends to generate patterns of considerable complexity and apparent relevance to many patterns we observe in natural systems. Pure RD is a minimal system that combines distinctions and recursion.

We have discussed in this essay the structure of RD and variants of it that are based on some or all its themes. There is a need in thinking to find simple basic principles and constituents from which all other apparent phenomena can be built. Here it is proposed that distinctions are such elementals. Distinctions escape the net of the conceptual exactly because the conceptual is based upon certain fundamental distinctions. Distinctions escape the simplicity of the physical for the same reasons. No one has ever isolated a distinction in nature that is not dependent upon some particular system of observations that gives rise to such distinctions for given observers.

The essential dialectic of recursive distinctioning is a round where meaning begets syntax and syntax begets meaning. That circularity is the basis of the world.

Meaning → Syntax
Syntax → Meaning

****

About the Author: Louis H. Kauffman is Emeritus Professor of Mathematics at the University of Illinois, Chicago. He has degrees from MIT and Princeton. He has 170 publications. He was the founding editor for the *Journal of Knot Theory and its Ramifications*, and he writes a column entitled "Virtual Logic" for the journal *Cybernetics and Human-Knowing*. He was president of the American Society for Cybernetics from 2005-2008. He introduced and developed the Kauffman Polynomial. He was the recipient of the 2014 Norbert Wiener Award of the American Society for Cybernetics. He is a Fellow of the American Mathematical Society.

Editors' Notes: This paper from Professor Louis Kauffman provides a further assessment of Joel Isaacson's work on recursive distinctioning. We hope that the reader finds this explanation both accessible and stimulating, and we are grateful for the contribution to this special issue honoring Isaacson's life and work. ***Gordon Arthur and Mark Wagner.***