COGNITIVE INTERPRETATION

William Bricken
January 1986

I've got some exciting (to me) cognitive applications in my thesis work. Last month, I decided to include/expand the chapter on cognitive modeling of error processes, so I'm working actively in that area now. It all ties in with a parallel (connectionist) representation and implementation of LoF. The general idea is to move Spencer-Brown's notation out of its linear format by mapping it onto graph theoretic networks. The mark of distinction becomes an empty node, whose only information content is in the connections to other nodes. Eg:

I:

```
                        ((a) (b))
```

II:

```
                        (       )              Extrude
                         ( ) ( )
                          a   b
```

III:

```
                        (       )              Link
                           / \
                         ( ) ( )
                         /     \
                        a       b
```

IV:

```
                            E                  Let ( ) be E (for Empty node)
                           / \
                          E   E
                         /     \
                        a       b
```

The network representation of AND. This structure needs a point of access. The observer, S, is represented explicitly, as the shallowest mark. Where the observer attaches his observation to the network determines its interpretation.

V:

```
                          S                    S sees the structure (a and b)
                           \
                            E
                           / \
                          E   E
                         /     \
                        a       b
```

A parallel, graphic representation has only one node for each variable. Since LoF requires two, it is *linear* in its representation. Eg:

Extract: (a) a ==> ( ) a

```
                   linear     network      transform

                      S           S               S
                     / \   =     / \     ==>     / \
                    E   a       E - a           E   a
                   /
                  a
```

Here's a remarkable consequence: Equivalence has three forms, only two of which are in LoF, because of the linearity of the notation we use.

A: (((a) b)((b) a))

```
                         E
                     /   |   \
                    a    |    E
                    |  S-E    |
                    E    |    b
                     \   |   /
                         E
```

B: (a b) ((a) (b))

```
                         E
                     /   |   \
                    E    |    E
                    |    S    |
                    a    |    b
                     \   |   /
                         E
```

C: equivalence network

```
                         S
                         |
                         E
                        / \
                      E - E
                     /     \
                    a       b
```

So, the notation is forcing not only duplicate (redundant) variable tokens, it is also forcing *redundant marks*. In form A, if we number the marks:

```
( ( (a) b ) ( (b) a ) )
1 2 3       4 5
```

The network representation shows that marks 2 = 5 and 3 = 4; they are the same distinction.

For a cognitive interpretation:

S: the environment of the structure (handles i/o)

E -- E: a network of *subconscious* processes (no label, no knowledge)

These structures are computable upon using a fully *parallel* process. The order of binding of variables determines whether or not the regime is functional (bound variables) or logical (variable constraints). An imaginary value at a labeled node propagates indeterminate values (errors) through the network.

I don't know whether or not you know these things already, since we are thinking along the same lines.