Boundary Logic

Advances in knowledge must necessarily appear to be unintelligible before their discovery and simple or obvious after their discovery.

Challenge

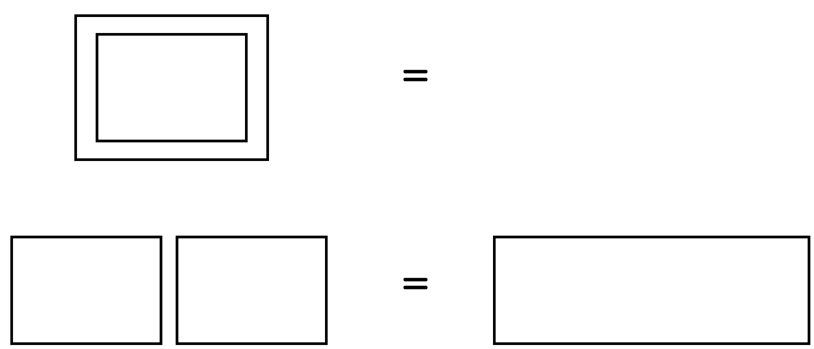
Computation and logic (Boolean algebra) are universally built on *binary* representations.

`0  1          True  False          Yes  No`

Is there a simpler approach? Can logic be expressed in a *unary* notation, with only one value?

Boundary Mathematics

Boundary math is the use of delimiting tokens, or containers, as both constants and functions. Here is an (pure math) example:



Common boundaries cancel.

Concepts

Boundary Token	an enclosure
Representational Space	the bounded space

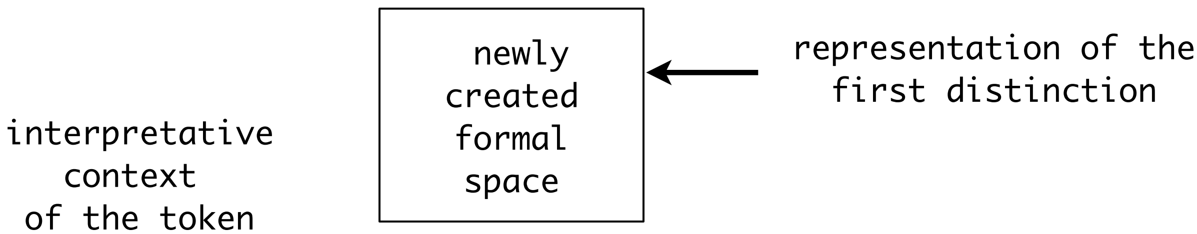
Two Voids

Absolute void	that which cannot be referred to without contradiction
Relative void	emptiness enclosed within a boundary

Constructing a Distinction

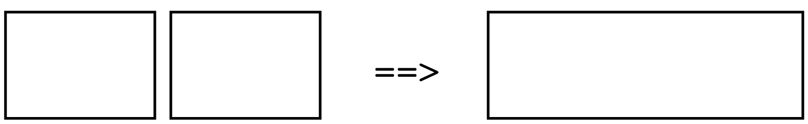
A ***universal distinction*** is first boundary we agree upon. In forming a first distinction, we construct three things simultaneously:

a **formal space** (inside)
a **representation** of the distinction (boundary, called ***mark***)
a **context** from which to interpret the distinction (outside)



Calling

Focus your attention on the outside, where you see the mark (the usual viewing point).
Call the boundary that you see a ***symbol***.

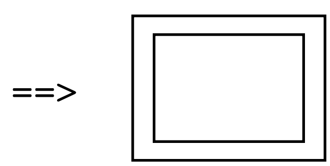


To call is to maintain perspective.

Calling is the rule of *invariance*. It is also is the rule of *naming*. The relationship between an object and its name is invariant.

Crossing

Focus your attention on the inside of a mark, where there is empty space.
Cross the boundary to the outside. Now you can see a ***mark***.



To cross is to change perspective.

Crossing is the rule of *variance*, and also a process of *change*. Crossing a boundary changes a value.

The Arithmetic of Boundaries

Calling `( ) ( )  =  ( )`

Crossing `( ( ) )  =`

Moving to Algebra

The ground value of boundary logic is represented by a ***delimiting token***, `( )`. The absence of that token can also be interpreted as a value outside of the representational system. If an equation holds for all ground values, it holds in general. Using this, we can construct algebraic truths from the cases of the arithmetic:

```
           DOMINION              INVOLUTION              PERVASION

       ( ) ( ) = ( )        (( ( ) )) = ( )        ( ( ) ) ( ) = ( )
       ( )     = ( )        ((     )) =            (     )     = ( )
thus
       ( )  A  = ( )        ((  A  )) =  A         (  A  )  A  = ( )
```

Boundary Rules of Transformation

The transformation rules of boundaries are:

Dominion `( ) A  =  ( )`

Dominion is the halting condition, when to stop.

Involution `((A)) = A`

Involution is how to remove excess boundaries.

Pervasion `A {A} = A { }`

Pervasion is how to remove excess replications of variables.

The curly braces in Pervasion stand in place of any and all inward structure. Each axiom suggests the same strategy for computation: ***erase irrelevant structure***

Algebraic transformation occurs by *substitution*. Any transform can be applied at any time and at any place in an expression without changing the value of that expression. Therefore, no transformation path changes the value of an expression. Some simplification paths may be longer and less efficient, but all lead to equivalent results.

Boundary Logic

Boundary logic uses a ***spatial representation*** of the logical connectives. Since *Calling* provides an object-oriented interpretation, and *Crossing* provides a process-oriented interpretation of the same mark, boundary forms can be evaluated using either an algebraic (match and substitute) approach or a functional (input converted to output) process.

Representation of logic and proof in spatial boundaries is new, and quite unfamiliar. Boundary logic is not based on language or on typographical strings, nor is it based on sequential steps. Boundary techniques are inherently parallel and positional. The meaning, or interpretation, of a boundary form depends on where the observer is situated. From the outside, boundaries are objects. From the inside, you cross a boundary to get to the outside; boundaries then are processes. This dramatically different approach (that is, permitting the observer to be an operator in the system) does not change the logical consequences or the deductive validity of a logical process.

Spatial representations do not have the concepts of associativity and commutativity. The base case is no representation at all, that is, the void has meaning in boundary logic. Simplification of logical expressions occurs by *erasing irrelevancies* rather than by accumulating facts.

Boundary Logic Representation

LOGIC	BOUNDARY	COMMENTS
False	<void>	No representation. Note: (( )) = <void>
True	( )	The empty boundary
A	A	Objects are labeled by tokens
not A	(A)	Negation is being on the inside
A or B	A B	Disjunction is sharing the same space
A and B	((A)(B))	Conjunction is a special configuration
if A then B	(A) B	Implication is separation by a boundary
A iff B	(A B)((A)(B))	Equality is spatially complex

In the above map from conventional logic to boundaries, many textual forms of logic condense into one boundary form. Note that the *parens*, ( ), is a linear, or one-dimensional, representation of a boundary. Circles and spheres are representations of boundaries in higher dimensions.

Boundary Logic Examples of Proof

To prove	*Transcribe and then apply the three axioms*	
A → A	(A) A	
	( ) A	pervasion
	( )	dominion
¬¬A = A	((A)) = A	
	A = A	involution
((A → B) ∧ A) → B	(((A) ((A) B))) B	
	(A) ((A) B) B	involution
	(A) ( ) B	pervasion of B and (A)
	( )	dominion
A ∧ B = ¬(¬A ∨ ¬B)	((A)(B)) = ((A)(B))	identity

The Natural Deduction Problem

Premise 1: If L then (if (not G) A)
Premise 2: If A then (if (D or not C) then G)
Premise 3: Not (if H then G)
Conclusion: Not (L and D)

Encode the logical connectives as boundaries, and simplify:

```
P1:     (L) ((G)) A      ==>   (L)  G  A
P2:     (A) (D (C)) G
P3:     ( (H) G )
Conc:   ( ((L) (D)) )    ==>   (L) (D)
```

Join all premises and conclusions into one form, using (P1 ∧ P2 ∧ P3) → Conc

```
( ((P1) (P2) (P3)) ) Conc   ==>   (P1) (P2) (P3) Conc     involution
```

Substitute the forms of the premises and conclusion, and reduce:

```
( (L) G A )  ( (A) (D (C)) G )  ( ((H) G) )  (L) (D)
( (L) G A )  ( (A) (D (C)) G )     (H) G     (L) (D)    involution
( (L)   A )  ( (A) (D (C))   )     (H) G     (L) (D)    pervasion of P
(       A )  ( (A) (D (C))   )     (H) G     (L) (D)    pervasion of (L)
(       A )  (     (D (C))   )     (H) G     (L) (D)    pervasion of (A)
(       A )         D (C)          (H) G     (L) (D)    involution
(       A )         D (C)          (H) G     (L) ( )    pervasion of D
                                                 ( )    dominion
                                                True    interpretation of ( )
```