GENERALIZATION OF PERVASION

William Bricken
November 1997

PERVASION: A (A B) = A (B)

"if it is on the outside of a distinction, it is arbitrarily on the inside."

1. Capital letters refer to arbitrary subgraph structures as well as atomic inputs. Example:

((a)(b)) (C ((a)(b))) = ((a)(b)) (C   )

2. Boundary transformations are deep:

A (B (C (A D))) = A (B (C (D)))

"the inside is arbitrarily deep"

3. Replication

A A = A

"the void is a zero order distinction"

4. Subsumption

(A) ((A C) B) = (A) (B)

Subsumption looks different in textual form, but it is still Pervasion with the image (aka enclosure) of the structure *A*, i.e. with *(A)*. The extraction removes the entire space containing A on the inside since the object on the outside is a bounded space (A).

5. Deep subsumption

(A) (B (C (A D))) = (A) (B (C   ))

6. Simple subsumption

(A) (A B) = (A)

7, Void Pervasion

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

8. Void subsumption

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

9. Deep subsumption with elements partially extracted at different depths. Example:

```
            (A B) (C (A (D (B E))))  =  (A B) (C (A (D      )))
```

10. Deep extraction and subsumption combined, across different depths:

```
      (A (B)) (C (A (D (B E))))  =  (A (B)) (C (A (D (  E))))

      (A B) (C (A (D (B E))))  =  (A B) (C (A (D      ))))

      (A (B C)) (A (B C D))  =  (A (B C)) (A (    D))
                             =  (A (B C D))    by distribution

      (A (B C D)) (A (B C))  =  (A (B C D))

      (A B C) (A (B C))  =  (A)
```

11. Insertion

This single-pass proof technique is a single algorithm which combines all of the less general interpretations of Pervasion.

12. Pervasion as a basis

```
            GIVEN:            A {A} = A { }
```

By degeneration:

```
                        A (A  ) = A ( )

                        A (A B) = A (B)

                        A  A    = A
```

These bases need the following axiomatic structures:

```
( A ( ))  = (( )) = <void>         Occlusion

  A ( )   =  ( )                   Dominion

((A)   )  =   A                    Involution
```

Deep Transformations

Like Pervasion and its sister Absorption, most boundary logic theorems are *deep*, in that they are insensitive to depth of nesting. The two exceptions are Distribution and Pivot, which are discussed in later sections.

Deep transformations greatly simplify proof and minimization by rendering intervening parens transparent while not increasing computational effort. Deep rules are extremely powerful and have no parallels in conventional logical theorems. Consider deep Absorption:

```
(A) {C (A B)} = (A) {C}                      ABSORPTION
```

The curly braces indicate that {C} can be any arbitrary intervening content or depth of nesting. Here are several examples of increasing apparent complexity. Forms deleted via Absorption are highlighted by spaces:

```
     (a) (c (d (e (a b)))
==>  (a) (c (d (e      ))

     (a b) (c (d (e (a b f g))))
==>  (a b) (c (d (e          )))

     (a b) (c (a d (e (b f))))
==>  (a b) (c (a d (e      )))
```

The last example illustrates another feature of deep rules: *constituent forms do not need to share the same space*. We can refine the definition of Absorption to emphasize this:

```
(A B) {C A {D (B E)}} = (A B) {C A {D}}       ABSORPTION
```

In the above equation, the {C} form indicates that the location of the embedded A form can be arbitrarily deep. The {D} form indicates that the location of the embedded B form can be arbitrarily deeper again than the A form. That is, the deleted (B E) form does not have to be in the same space as the embedded A. This is substantially different than all conventional equational or logical transformations, which express rules for expressions only at the same level of depth. Deep rules in effect disassociate operators from their immediate arguments.

A deep rule can be proved as a theorem by successive application of shallow rules. Consider a proof of deep Absorption, expressed without depth braces:

```
(A B) (        C (        A (      D (      B E))))
(A B) (^(A B)^ C (        A (      D (      B E))))         ins (A B)
(A B) (^(A B)^ C (^(A B)^ A (      D (      B E))))         ins (A B)
(A B) (^(A B)^ C (^(  B)^ A (      D (      B E))))         per A
(A B) (^(A B)^ C (^(  B)^ A (^(B)^ D (      B E))))         ins (B)
(A B) (^(A B)^ C (^(  B)^ A (^(B)^ D (^(B)^ B E))))         ins (B)
(A B) (^(A B)^ C (^(  B)^ A (^(B)^ D (^( )^ B E))))         per B
(A B) (^(A B)^ C (^(  B)^ A (^(B)^ D            )))         occ
(A B) (^(A B)^ C (^(  B)^ A (      D            )))         per (B)
(A B) (^(A B)^ C (^(A B)^ A (      D            )))         ins A
(A B) (^(A B)^ C (        A (      D            )))         per (A B)
(A B) (        C (        A (      D            )))         per (A B)
```

This approach is a fundamental miscomprehension of boundary logic theory in two distinct ways. Primarily it confuses a theorem with an algorithm which implements the theorem on a particular class of machines.

The first problem with this *step-wise proof* is that the carated form ^(A B)^ is virtual, it is void-equivalent and does not exist in the form. Thus it need not be Occluded to be erased. Virtual forms are simply abandoned into <void> after they have served their reduction purpose. This point is illustrated by changing the steps of the above proof which follow Occlusion:

```
(A B) (^(A B)^ C (^(  B)^ A (^(B)^ D            )))         occ
(A B) (        C (        A (      D            )))         virtual
```

Second, deep rules are independent of depth of nesting and do not need to incorporate boundary crossing steps. Parens are *transparent* to the outside, they are computationally semipermeable. The intermediate copies of (A B) should not be recorded, since only one replicate need exist to achieve the goal of insertion. Thus, the appropriate proof of deep Absorption calls upon deep Pervasion directly:

```
(A B) (        C (        A (      D (        B E))))
(A B) (        C (        A (      D (^(A B)^ B E))))       ins (deep)
(A B) (        C (        A (      D (^(  B)^ B E))))       per A (deep)
(A B) (        C (        A (      D (^(   )^ B E))))       per B
(A B) (        C (        A (      D              )))       occ
```

This understanding permits the curly braces generalization to Absorption, where curly braces represent any number of levels of intervening parens, while the label attached to the curly braces represents any arbitrary form(s):

```
(A B) {C         (        A {D       (        B E)})}
(A B) {C         (        A {D                    })}       abs
```