A SOUNDNESS PROOF OF C.S. PEIRCE’S ALPHA EXISTENTIAL GRAPHS USING TRUTH TREES, WITH AN APPENDIX ON THE “ALPHA” ENTITATIVE GRAPHS

by

RICHARD SCOTT SHEDENHELM

(Under the Direction of O. Bradley Bassler)

ABSTRACT

This work represents a marriage of Peirce’s syntactic rules for performing inferences in the Alpha Existential Graphs with the recent technique of truth trees. The vitality of this marriage is brought to light by demonstrating the soundness of the Alpha system in a much more diagrammatic way than has ever before been accomplished. While the use of truth trees to work with properties such as soundness and completeness is not new in this work, the employment of a graphical form of the trees is novel, especially in the context of Peirce’s graphical logic. Finally, I suggest how the same method would apply to Peirce’s earlier system, the Entitative Graphs.

INDEX WORDS: Charles Sanders Peirce, Graphical logic, Existential graphs, Entitative graphs, Truth trees, Soundness

A SOUNDNESS PROOF OF C.S. PEIRCE'S ALPHA EXISTENTIAL GRAPHS USING TRUTH TREES, WITH AN APPENDIX ON THE "ALPHA" ENTITATIVE GRAPHS

by

RICHARD SCOTT SHEDENHELM

A.B., The University of Georgia, 1995

A Thesis Submitted to the Graduate Faculty of The University of Georgia in Partial Fulfillment of the Requirements for the Degree

MASTER OF ARTS

ATHENS, GEORGIA

2002

A SOUNDNESS PROOF OF C.S. PEIRCE'S ALPHA EXISTENTIAL GRAPHS USING TRUTH TREES, WITH AN APPENDIX ON THE "ALPHA" ENTITATIVE GRAPHS

by

RICHARD SCOTT SHEDENHELM

Major Professor:	O. Bradley Bassler
Committee:	Robert G. Burton Charles B. Cross

Electronic Version Approved:

Maureen Grasso
Dean of the Graduate School
The University of Georgia
December 2002

Acknowledgments

This project has been a five-year endeavor of love and occasional hate. My foremost debt is to O. Bradley Bassler, who first suggested my research into Peirce's graphical logic and has since been a patient and searching critic of my work. When I reflect upon this project, I think of the quotation of Thomas Edison: "Results! Why, man, I have gotten a lot of results. I know several thousand things that won't work. I found five thousand ways how not to make a light bulb." In similar vein, I have developed numerous ways how *not* to prove the soundness of the Alpha graphs. If there really is enlightenment at the end of this work, then I have to thank Dr. Bassler for keeping the power on.

With gratitude I thank the other members of my committee, Drs. Robert G. Burton and Charles B. Cross, for their patience in awaiting the completion of this work and their counsel that mathematical precision is not enough for a project of this sort: it ought to include some intelligible English as well.

Furthermore, I thank Sun-Joo Shin for her insight and encouragement. A recent work of hers, although not directly concerned with the soundness of the Existential Graphs, has suggested different methods for reading the graphs that helped the present project in formulating semantic decomposition principles.

Finally I would like to thank my wife Laura and good friend William T. Murray for enduring five years of my ravings about "concentric hockey pucks (Ketner's analogy for visualizing graphical cuts). I hope this work will justify the ravings.

Table of Contents

I. Introduction

In late 1896, C.S. Peirce began work on a system of graphical logic that he called "Existential Graphs." He developed the basic conventions of this system while reviewing the draft pages of a *Monist* article that was published in January 1897. (Cf. Ketner, 1987, for details.) That *Monist* article contained an earlier and related system, the "Entitative Graphs." Peirce regarded his later system as a vast improvement upon the earlier, and he continued work on it until his death, motivated by topological and metaphysical concerns. (Zeman, 1964, 1968; Kent.) The Existential Graphs consist of three main parts, the Alpha, Beta, and Gamma graphs. The Alpha graphs correspond to propositional logic, Beta graphs to predicate logic, and the Gamma graphs to modal and metalogic.

The method of truth trees to deal with semantic entailment was first proposed by Richard Jeffrey in 1967, applied to relevance logic by Michael Dunn in 1975, and has since become a common tool in introductory symbolic logic texts. On the whole, this work represents a marriage of Peirce's syntactic rules for performing inferences in the Alpha Existential Graphs with the recent technique of truth trees. The vitality of this marriage is brought to light by demonstrating the soundness of the Alpha system in a much more diagrammatic way than has ever before been accomplished. While the use of truth trees to work with properties such as soundness and completeness is not new in this work, the employment of a graphical form of the trees is novel, especially in the context of Peirce's graphical logic. Although previous soundness proofs have been offered for the Alpha graphs (Roberts, 1973, Stewart, and Hammer), the Beta graphs (Roberts, 1973), and the Gamma graphs (Butterworth), none of them employed truth trees. Trees seem to present the semantics of Peirce's graphical logic in a manner consistent with his intention,

> What we have to do, therefore, is to form a perfectly consistent method of expressing any assertion diagrammatically. The diagram must then evidently be something that we can see and contemplate. (4.430)

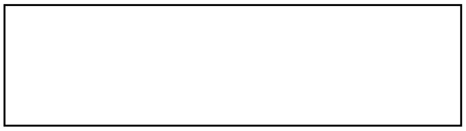
In sections II and III, I identify the syntactic constraints on writing the graphs and some terminology used in referring to the graphs. In section IV, based on Peirce's conventions for reading the graphs, I illustrate the three fundamental types of diagrams that will occur in the decomposition trees, viz., the types of diagrams representing conjunction, disjunction, and double negation. In section V, I define the four kinds of decomposition allowed in the diagrammatic truth trees. In section VI, I demonstrate a number of useful properties relating to the graphs lying within a diagram subject to decomposition and how the graphs behave in the resulting diagrams. Section VII enumerates further conventions and definitions for the truth trees which set a grounding for the proofs of various tree theorems in Section VIII. After naming Peirce's five Alpha inference rules in section IX, I go on in the following section to prove the soundness of the Alpha system. In section XI, I offer a conjecture as to what the inference rules in Entitative Graphs should be, how inferences look when compared to Existential Graphs, and how we might go about showing that the rules of the Entitative Graphs are sound.

Finally in section XII, I suggest a procedure for translating statements in algebraic logic into Existential Graphs.

II. Basic Syntax

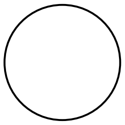
1. Graphical primitives: The graphical primitives are the fundamental constituents out of which the logical graphs are constructed. Rules for the construction of these graphs will be given in II.2-3 below. Later, in section IV, we shall stipulate that the Roman letters stand in place of declarative sentences.

Sheet of Assertion (SA):



Capital Roman letters: A, B, C, ...

Cuts:



2. Area definitions

a) The border of either the SA or a cut is represented by the line forming the rectangle or circle, respectively.

b) The region inside the border of the SA is the area of the SA.

c) The region inside the border of a cut is the area of the cut.

3. Formation rules for graphs

a) Non-overlapping rule:

i) No border of the SA or cut is to intersect any other border or capital Roman letter.

ii) No capital Roman letter is to intersect another capital Roman letter.

iii) This rule restricts the use of any further formation rules.

b) Where α is any capital Roman letter, the letter denoted by α may be written in the area of the SA.

c) Where β is any cut, the cut denoted by β may be written in the area of the SA.

d) Where β is any cut, the cut denoted by β may be written in the area of another cut.

e) Where α is any capital Roman letter, the letter denoted by α may be written in the area of any cut.

f) No other graph is well-formed unless formed by finitely many applications of rules a-e.

III. Descriptive Syntax

In order to be able to specify the particular locations within a graph, we develop a way of referring to the various levels within a system of cuts. The following definitions will allow us to refer to and distinguish diagrams that represent conjunction, disjunction, double negation, and simple assertion or denial, as well as certain important systems of cuts.

1. Area magnitude of cuts

a) Let the number 0 be the area magnitude assigned to the area of the SA.

b) Let the number 1 be the area magnitude assigned to the area of any cut such that no points on the border of the cut lie in the area of any other cut.

c) Let the number 2 be the area magnitude assigned to the area of any cut such that all the points on the border of the cut lie in the area of a cut of magnitude 1.

d) In general, let the number *n* be the area magnitude assigned to the area of any cut such that all the points on the border of the cut lie in the area of a cut of magnitude *n*-1.

2. Definition of a diagram: Let a diagram be a single instance of the SA with whatever, if any, finite number of additional well-formed graphs are written in the area of the SA.

3. Definition of "lying in the area of the SA":

A graph α of a diagram is lying in the area of the SA if and only if there exists no cut around α.

Hence, a graph α is lying in the area of the SA if and only if α is lying in an area of magnitude 0.

4. Atomic graph: Any graph consisting of:

a) just the blank SA with no graphs lying in the area of the SA, or

b) just one capital Roman letter, or

c) just one cut that contains only one Roman letter within its area, or

d) just one cut that has no Roman letters or cuts within its area

5. Atomic diagram: Any diagram consisting of the SA with at most one additional well-formed atomic graph written in the area of the SA.

6. Double cut: Two cuts α, β such that:

a) one cut α with area magnitude *n* has all the points of its border lying in another cut β that has area magnitude *n*-1, *n* = 1, 2, 3, . . . , and

b) no other graph lies in the area of β except: cut α, with whatever graphs are lying in the area of the cut α.

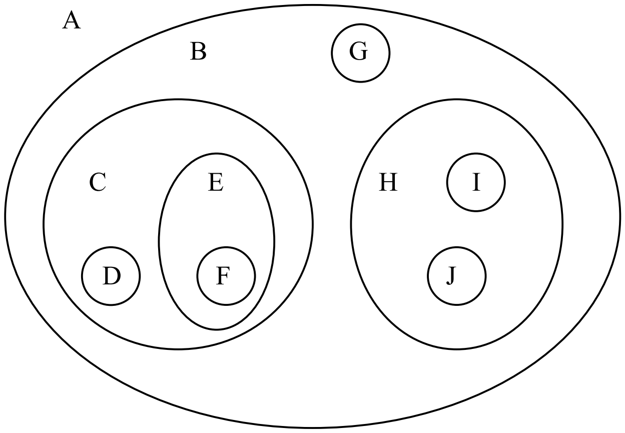
c) Let cut α be called the "inner cut" of the double cut. Let cut β be called the "outer cut" of the double cut.

7. Nest of cuts: "A series of cuts each enclosing the next one." (Roberts, 1973, p. 153) As Peirce says: "A nest is any series of cuts each enclosing the next one." (Ms 693, p.

292, as quoted in Roberts, 1973, p. 37) And further, from Ms 650, as quoted in Roberts, 1973, p. 37:

> If there be a collection, i.e., a definite, and individual plural of cuts, of which one is placed in the sheet of assertion, and another encloses no cut at all, while every other cut of the series has the area of another cut of this collection for its place, and has its area for the place of still a third cut of this collection, then I call that collection a *nest*, and the areas of its different cuts *its successive areas*, and I number them originally from the sheet of assertion as *origin*, or zero, with an increase of unity for each passage across a cut of the nest inwards that one can imagine some insect to make if it never passes out of an area that it has once entered. For example, in Fig. 1 there are five nests as follows:

Fig. 1

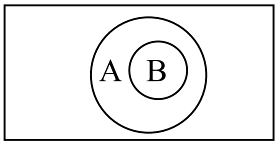


1. One of 5 areas, or 4 cuts; A-B-C-E-F,
2. A-B-C-D,
3. Three of 4 areas or 3 cuts each; A-B-H-I,
4. A-B-H-J,
5. One of 3 areas, or 2 cuts; A-B-G

8. Place of a cut or graph: The area on which a cut is made or a graph is scribed. Let the place by symbolized by two pairs of square parentheses in italics, so that *[[*P*]]* denotes the place of P.

9. Gödel numbering for the graphs: We will need a method to ensure that the decomposition procedures codified in section V can be applied in a mechanical way. To this end, we state a Gödel-numbering algorithm.

In 4.378 Peirce employs a method, which I shall term ‘linear notation’, for typing the graphs using typographical characters. With a slight modification, suggested by Hammer, I’ll employ it here. For each sentence letter, use the same sentence letter. For a juxtaposition of two or more sentence letters lying in the same area, concatenate them. For a graph surrounded by a cut, enclose the graph by a pair of parentheses. For example, for the diagram



we can type ‘(A(B))’. As Peirce notes, we can also type ‘((B)A)’ for the same diagram.

Algorithm:

a) Put the graph into linear notation:

i) cuts represented by a pair of matching parentheses.

ii) juxtaposition on same area represented by concatenation.

b) Assign Gödel number to linear notation:

i) for each ‘(’: 1

ii) for each ‘)’: 2

iii) for each sentence letter, following standard alphabetical order:

‘A’: 3

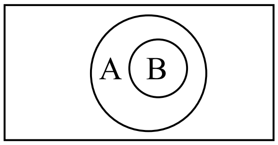
‘B’: 4

‘C’: 5

etc.

c) Reading left to right the linear notation of *n* symbols, associate the unique number that is the product of the first *n* primes in order of magnitude, each prime being raised to a power equal to the Gödel number of the corresponding symbol.

For example, consider the following diagram and how to determine the Gödel numbering.



(A (B))

$$2^1 \times 3^3 \times 5^1 \times 7^4 \times 11^2 \times 13^2$$

or, using the alternative arrangement:

(B (A))

$$2^1 \times 3^4 \times 5^1 \times 7^3 \times 11^2 \times 13^2$$

Since we need a unique way to assign Gödel numbering, let the lowest possible Gödel number (LPGN) for a graph be the minimum product for all the possible typological arrangements that can be made for a graph. We will use the LPGN to order the graph in such a way that the decomposition rules in section V may be applied in a determinate fashion.

IV. Peirce's Semantics for the Graphs

In various places in his writings concerning the Existential Graphs, Peirce states the principles by which the graphs are to represent statements, including affirmations, denials, conjunctions, and disjunctions. The interested reader may compare the conventions stated here for the Existential Graphs with Appendix 1, which discusses the conventions of the earlier Entitative Graphs.

The following quotations come from Peirce's discussion of the Existential Graphs:

> A sep, or lightly drawn oval, when unenclosed is with its contents (the whole being called an enclosure) a graph, entire or partial, which precisely denies the proposition which the entire graph within it would, if unenclosed, affirm. Since, therefore, an entire graph, by the above principles, copulatively asserts all the partial graphs of which it is composed . . . it follows that an unenclosed enclosure disjunctively denies all the partial graphs which compose the contents of its sep. . . . Consequently, if an enclosure is oddly enclosed, its evenly enclosed contents are copulatively affirmed; while if it be evenly enclosed, its oddly enclosed contents are disjunctively denied. (4.474)
>
> Two cuts one of which has the enclosure of the other on its area and has nothing else there constitute a double cut. (4.414)
>
> . . . [Two] negatives make an affirmative . . . (4.379)

In order to clearly articulate Peirce's conventions, I shall take the following principles to be how the graphs should be read:

1. The capital Roman letters will stand in place of declarative statements.

2. A single graph lying in the area of the SA is to be read as the assertion of the truth of the proposition expressed by the graph.

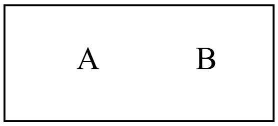
3. A juxtaposition of graphs lying in the area of the SA is to be read as the conjunction of the graphs.

4. A cut around a graph is to be read as the denial of the truth of the proposition expressed by the graph enclosed by the cut.

5. A cut around a juxtaposition of graphs is to be read as the denial of the conjoined truth of the graphs juxtaposed within the enclosure of the cut.

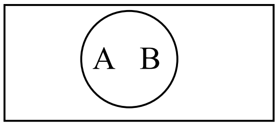
6. A double cut around a graph is to be read as the double negation of the graph enclosed within the inner cut.

For example, the following diagram expresses a conjunction:



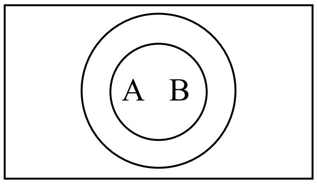
Read as: "Both A and B are true"

The following diagram expresses a disjunction:



Read as: "It is false that both A and B are true"
or: "Either A is false and/or B is false"

The last example is a diagram expressing double cut, i.e., double negation:



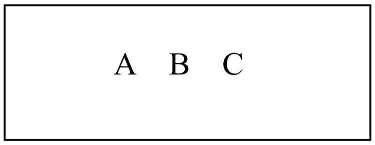
Read as: "It is false that it is false that both A and B are true"

V. Decomposition Principles

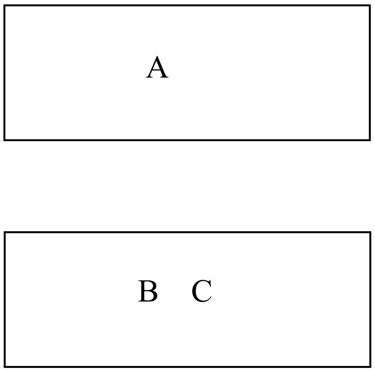
In order to construct truth trees for the diagrams, we need guidelines for deconstructing non-atomic diagrams. In most presentations of truth trees, there are nine rules for decomposition: The first four rules correspond to statements whose main connective is one of the four binary operators (e.g., &, ∨, ⊃, ≡), the second four to negations of statements whose main connective is one of the four binary operators, and the ninth for double negations.

In contrast, given the conventions for reading the diagrams, we can be more economical. We need only three decomposition rules, one for conjunctions, a second for disjunctions, and a third for double negations. However, as I shall discuss immediately below at V.4, we need an additional rule--a variation of disjunction--to cover certain peculiar kinds of semantic entailments.

1. The first kind of diagram we want to decompose is one that represents a conjunction, that is, a set of two or more graphs lying in the area of the SA. For example, the following diagram represents the conjunction of statements A, B, and C:



We want to decompose the previous diagram into two diagrams, each of which contain fewer graphs than the previous one. For example:



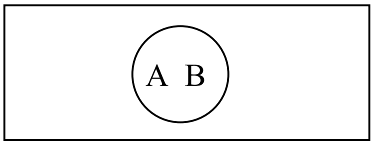
Furthermore, we want to specify in a determinate manner exactly which graphs from the original diagram will occur in the two resulting diagrams. The technique of Gödel numbering will help us to accomplish this need.

Conjunction Decomposition (CD): Let diagram D consist of the SA with set A of two or more additional graphs, A = {α, β, ...}, lying in the area of the SA. Let α be a graph in set A that has the least LPGN. If two or more graphs in set A have the same least LPGN, then let any one of them be graph α. Let B be the set of graphs B = A\{α}. Then:

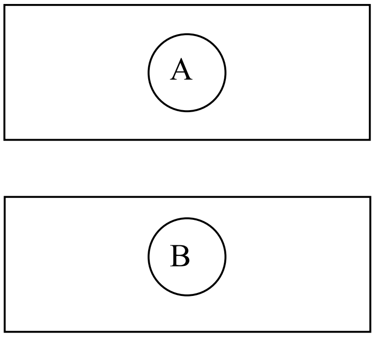
a) D' may be written consisting of the SA with graph α in the area of the SA, and

b) D" may be written consisting of the SA with each of the graphs in B written in the area of the SA.

2. The second kind of diagram we want to decompose is one that represents a disjunction, that is, a set of two or more graphs lying in the area of a cut lying in the area of the SA. For example, the following diagram represents the disjunction of statements not-A and not-B:



As with diagrams representing conjunctions, we want to decompose the previous diagram into two diagrams, each of which contain fewer graphs than the previous one. For example:



And again, as before, we want to specify in a determinate manner exactly which graphs from the original diagram will occur in the two resulting diagrams. So we shall again use the technique of Gödel numbering to accomplish this need.

Disjunction Decomposition (DD): Let diagram D consist of:

a) the SA with only one graph α lying in the area of the SA, and

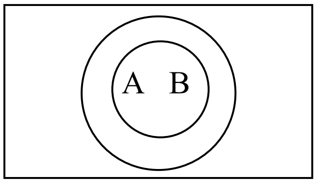
b) graph α consists of a cut with set A of two or more graphs, $A = \{\beta, \gamma, ...\}$, lying in the area of α. Let the graph β be a graph in set A that has the least LPGN. If two or more graphs in set A have the same least LPGN, then let any one of them be graph β. Let B be the set of graphs $B = A\backslash\{\beta\}$.

Then:

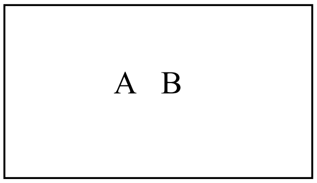
c) D' may be written consisting of the SA, one cut lying in the area of the SA, and graph β lying in the area of the cut, and

d) D" may be written consisting of the SA, one cut lying in the area of the SA, and each of the graphs in set B written in the area of the cut.

3. The diagram representing a double negation consists of one or more graphs enclosed within a double cut where no Roman letters or cuts are written outside of the double cut. For example, the diagram



represents the double negation of A-and-B. We want a decomposition procedure that eliminates the double cut, resulting in, say, the following diagram:



Double Cut Decomposition (DCD): Let diagram D have a single cut having area magnitude 1. If this cut is the outer cut of a double cut, then D' may be written consisting of the SA and all other tokens in D *except* the inner and outer cuts that had constituted the double cut in D.

4. As I will explain later, in section VII.4, when we use the decomposition principles to form "coupled trees" (cf. VII.2.m), we may require a certain variation of DD that does

not involve the goal of simplifying other diagrams. Whereas in CD, DD, and DCD, the goal is to construct new diagrams, each of which contains fewer graphs than the original diagram, our additional procedure will not exist for that purpose. Indeed, the diagrams resulting from the procedure may not contain any of the graphs that had occurred in other diagrams within the tree. In short, we need a decomposition procedure that captures the permission to assert the law of excluded middle, whenever we please.

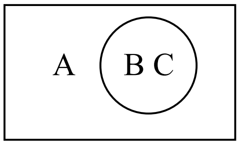
Free Disjunction Decomposition (FDD): Let D be a diagram and α any graph. Then D' may be written consisting of the SA and any graph α lying in the area of the SA and D" may be written consisting of the SA, one cut lying in the area of the SA, and a token of graph α written in the area of the cut.

Note: In section VII, we will introduce tree principles for the applications of these decomposition principles within tree diagrams. These principles will, in particular, allow us to represent the fact that in applications of CD we must take the conjunction of the diagrams D' and D" and in applications of DD we must take the disjunction of D' and D".

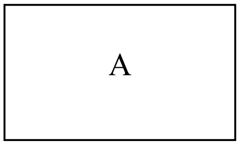
VI. Decomposition Theorems

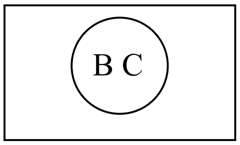
The following theorems will give us the foundation to describe in a precise way how CD, DD, and DCD act on tokens of graphs in relation to the original diagram subject to these decomposition procedures. In particular, we will be able to identify how these three decomposition procedures affect the area magnitude of the tokens and the number of tokens within the diagrams.

1. In a CD-type diagram, we have two or more graphs lying in the area of the SA, viz., in an area of magnitude 0. For example, in the following diagram, we have the graphs representing A and the negation of B-and-C lying in the area of the SA:



We want to identify what effect CD has to the area magnitude of the token graphs that exist in the resulting diagrams:





Area magnitude constancy lemma for CD: the graphs α, β, ... which were lying in an area of magnitude 0 before CD (viz., within the area of the SA), shall lie in an area of magnitude 0 (viz., the SA) after CD.

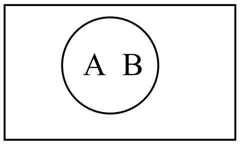
Proof:

CD assumes that diagram D shall consist of the SA with two or more additional graphs, A = {α, β, ...}, lying in the area of the SA. Hence, by the definition of "lying in the area of the SA," CD assumes that diagram D shall consist of the SA with two or more additional graphs A = {α, β, ...} lying in an area of magnitude 0.

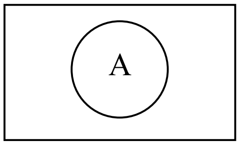
Furthermore, the CD rule stipulates that after CD, D' shall be written consisting of the SA with graph α in the area of the SA, and that D'' shall be written consisting of the SA with each of the graphs in B = A\{α} written in the area of the SA. Hence, by the definition of "lying in the area of the SA," CD stipulates that after CD α shall lie in an area of magnitude 0, and each of the graphs in B shall lie in an area of magnitude 0.

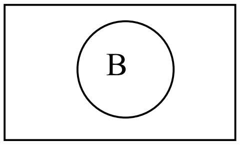
Corollary: For all graphs lying in an area of magnitude *n* before CD (*n* = 0, 1, 2, . . .), they shall lie in an area of magnitude *n* after CD.

2. In a DD-type diagram, we have two or more graphs lying in the area of a cut, which cut lies in the area of magnitude 0. For example, in the following diagram, we have the graphs representing A and B lying in the area of a cut that rests in the area of the SA:



We want to identify what effect DD has to the area magnitude of the token graphs that exist in the resulting diagrams:





Area magnitude constancy lemma for DD: The graphs β, γ, ... which were lying in an area of magnitude 1 before DD, shall lie in an area of magnitude 1 after DD.

Proof:

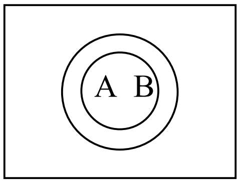
DD assumes that diagram D shall consist of the SA with just graph α lying in the area of the SA and that graph α consists of a cut with set A = {β, γ,...} lying in the area of α. By the definition of "lying in the area of the SA," DD assumes that diagram D shall consist of the SA with just graph α lying in an area of magnitude 0. But then, each of the graphs in A = {β, γ,...} is lying in an area of magnitude 1.

Furthermore, the DD rule stipulates that after DD, D' shall be written consisting of the SA, one cut lying in the area of the SA, and graph β lying in the area of the cut. But then, by the definition of "lying in the area of the SA," the cut around β is lying in an area of magnitude 0. Hence, β lies in an area of magnitude 1.

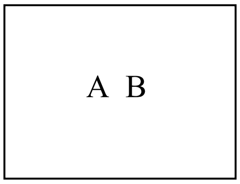
Finally, the DD rule stipulates that after DD, D" shall be written consisting of the SA, one cut lying in the area of the SA, and each of the graphs in the set B = A\{β} lying in the area of the cut. But then, by the definition of "lying in the area of the SA," the cut around the set of graphs B = A\{β} is lying in an area of magnitude 0. Hence, each of the graphs in B = A\{β} is lying in an area of magnitude 1.

Corollary: For all graphs lying in an area of magnitude *n* before CD (*n* = 0, 1, 2, . . .), they shall lie in an area of magnitude *n* after CD.

3. In a DCD-type diagram, we have two or more graphs lying in the area of the inner cut of a double cut, the outer cut of which lies in the area of the SA. For example, in the following diagram, we have the graphs representing A and B lying in the area of the inner cut:



We want to identify what effect DCD has to the area magnitude of the token graphs that exist in the resulting diagrams:



Area magnitude reduction lemma for DCD: for all graphs lying in the area of magnitude 2 before DCD (viz., within the inner cut), they shall lie in an area of magnitude 0 (viz., the SA) after DCD.

Proof:

The graphs A = {α, β, ...} lying in the area of magnitude 2 before DCD were in an area of magnitude 2 just because all the points on the border of the closest cut (the inner cut) surrounding all the graphs in A were lying in an area of a cut (the outer cut) of magnitude 1. By DCD, D' shall contain all the tokens in D except the inner and outer cuts that had constituted the double cut in D. Hence, all the graphs in A that had been in the area of magnitude 2 before DCD--i.e., that had been surrounded by the double cut--shall not be surrounded by the double cut after DCD. Since the outer cut of the double cut was lying in the area of the SA prior to DCD, the graphs in A shall lie in the area of the SA after DCD. Hence, the graphs in A shall lie in an area of magnitude 0 after DCD.

Corollary: For all graphs lying in an area of magnitude $n+2$ before DCD ($n = 0, 1, 2, \ldots$), they shall lie in an area of magnitude n after DCD.

4. Token Cardinality. In order to demonstrate that a finite number of decomposition procedures will accomplish the decomposition of a diagram and all resulting diagrams to atomic diagrams, we need to identify how CD, DD, and DCD act to reduce the number of graph tokens in each of the resulting diagrams. We employ a notion of graph token cardinality to specify the number of graphs within a diagram. First we shall illustrate the principles in a schematic way. Then, in 5-8 below, we shall prove three lemmas pertaining to token reduction.

For a well-formed graph α, let

$C = \{x : x \text{ is a cut written in } \alpha\}$

$L = \{y : y \text{ is a token of a capital Roman letter written in } \alpha\}$

For a well-formed graph α that does *not* include the SA:

$\#_{\alpha}(C)$ = the number of cuts written in α

$\#_{\alpha}(L)$ = the number of tokens of capital Roman letters written in α

$\#\alpha = \#_{\alpha}(C) + \#_{\alpha}(L)$.

Let #SA = 1; for a well-formed graph α that does include the SA:

$\#\alpha = \#_{\alpha}(C) + \#_{\alpha}(L) + 1$.

Note that for all graphs α, $\#\alpha \geq 1$.

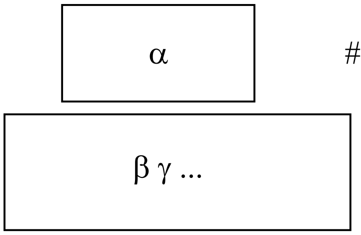
a) For a CD-type diagram D, that is, a diagram to which CD can be applied, with graphs $\alpha, \beta, \gamma, \ldots$ lying in an area of magnitude 0,

$$\#D = 1 + \#\alpha + \#\beta + \#\gamma \ldots$$

Furthermore, we can schematically write an instance of CD as follows. First we represent the original diagram:

$$\boxed{\quad \alpha\ \beta\ \gamma \ldots \quad} \qquad \#D = 1 + \#\alpha + \#\beta + \#\gamma \ldots$$

Next, we write the two decomposition diagrams one on top of the other as follows:



$\#D' = 1 + \#\alpha$

$\#D'' = 1 + \#\beta + \#\gamma \ldots$

$\#D' = \#D - (\#\beta + \#\gamma \ldots) = 1 + \#\alpha$

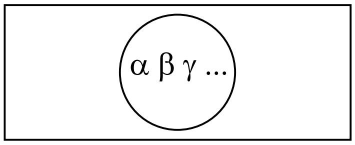
$\#D'' = \#D - \#\alpha = 1 + \#\beta + \#\gamma \ldots$

Note that #D', #D" < #D.

b) For a DD-type diagram D, that is, a diagram to which DD can be applied, with graphs α, β, ... lying in an area of magnitude 1,

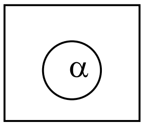
$$\#D = 2 + \#\alpha + \#\beta + \ldots$$

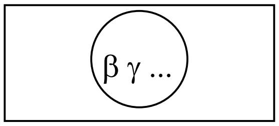
Furthermore, we can schematically write an instance of DD as follows. First we represent the original diagram:



$\#D = 2 + \#\alpha + \#\beta + \#\gamma \ldots$

Next, we write the two decomposition diagrams one next to the other as follows:





$\#D' = 2 + \#\alpha$ $\qquad\qquad$ $\#D'' = 2 + \#\beta + \#\gamma \ldots$

$\#D' = \#D - (\#\beta + \#\gamma \ldots) = 2 + \#\alpha$

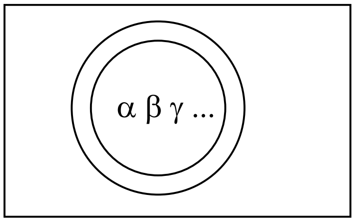
$\#D'' = \#D - \#\alpha = 2 + \#\beta + \#\gamma \ldots$

Note that $\#D', \#D'' < \#D$.

c) For a DCD-type diagram D, that is, a diagram to which DCD can be applied, with graphs α, β, ... lying in an area of magnitude 2,

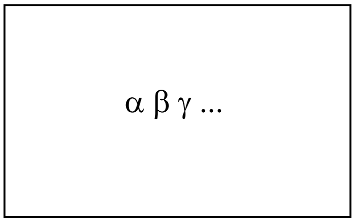
$$\#D = 3 + \#\alpha + \#\beta + \ldots$$

Furthermore, we can schematically write an instance of DCD as follows. First we represent the original diagram:



$\#D = 3 + \#\alpha + \#\beta + \#\gamma \ldots$

Next, we write the one decomposition diagram as follows:



$\#D' = 1 + \#\alpha + \#\beta + \#\gamma \ldots$

$\#D' = \#D - 2$

Note that $\#D' < \#D$.

The above schematic illustrations can be articulated and proved formally for each decomposition principle CD, DD, and DCD. This we do in 5-8 below.

5. **Token reduction lemma for CD**: Let diagram D have *n* tokens. Then after CD, D' shall have fewer than *n* tokens and D" shall have fewer than *n* tokens.

Proof: By CD, diagram D consists of the SA with set A of two or more additional graphs, A = {α, β, ...}, lying in the area of the SA. D' is written consisting of the SA with graph α in the area of the SA, viz., D' has fewer tokens than D. D" is written consisting of the SA with each of the graphs in B = A\{α} written in the area of the SA, viz., D" has fewer tokens than D. (See above, VI.4a)

6. **Token reduction lemma for DD**: Let diagram D have *n* tokens. Then after DD, D' shall have fewer than *n* tokens and D" shall have fewer than *n* tokens.

Proof: By DD, diagram D consists of the SA with only one graph α lying in the area of the SA, where graph α consists of a cut with set A or two or more graphs, A = {β, γ, ...}, lying in the area of α. D' is written consisting of the SA, one cut lying in the area of the SA, and graph β lying in the area of the cut, viz., D' has fewer tokens than D. D" is written consisting of the SA, one cut lying in the area of the SA, and each of the graphs in set B = A\{β} written in the area of the cut, viz., D" has fewer tokens than D. (See above, VI.4b)

7. **Token reduction lemma for DCD**: Let diagram D have *n* tokens. Then, after DCD, D' shall have *n*-2 tokens.

Proof: By DCD, D' shall consist of the SA and all other tokens in D except the inner and outer cuts that had constituted the double cut in D. But both the inner and outer cuts are tokens. Hence, D' shall have two fewer tokens than D. (See above, VI.4c)

We summarize results 5, 6, 7, in

8. **Diagram Reduction Theorem**: By each application of the decomposition principles upon a given diagram D,

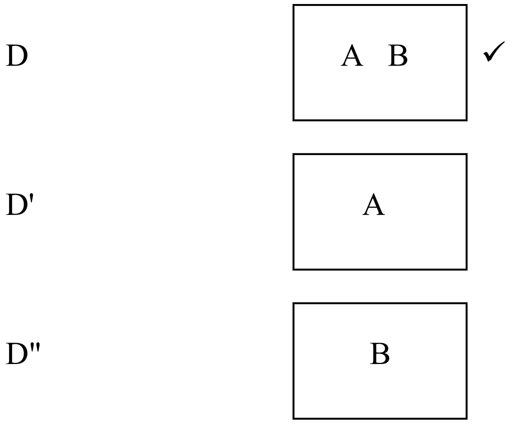
a. For each resulting diagram(s) there exist fewer tokens than in D.

b. For the diagram resulting from an application of DCD, it contains graphs lying in areas two less than in D.

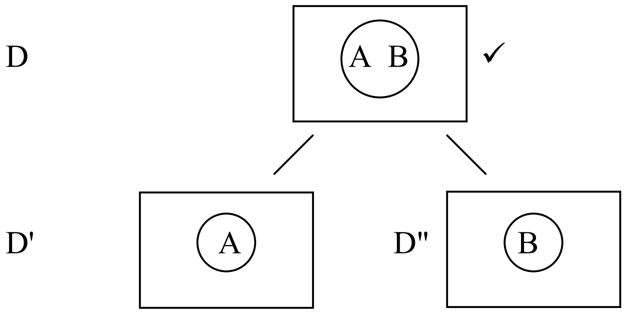
VII. Tree Principles

1. Conventions for the trees:

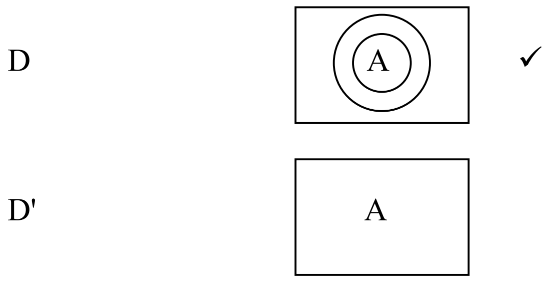
a) When applying CD, let diagram D be written just above D' and D' be written just above D". Add a check mark next to D. For example:



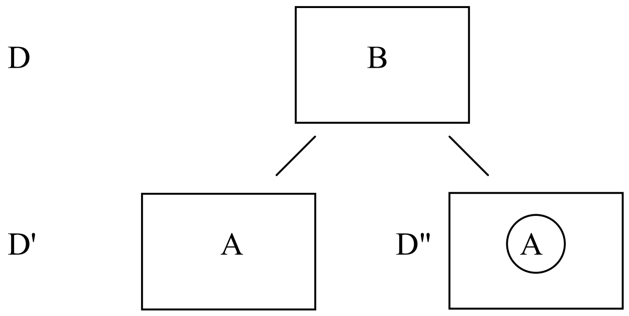
b) When applying DD, let diagram D be written just above D' and D" so that D' is juxtaposed to the left of D". Add a check mark next to D. Insert a line joining D and D' and insert a line joining D and D". For example:



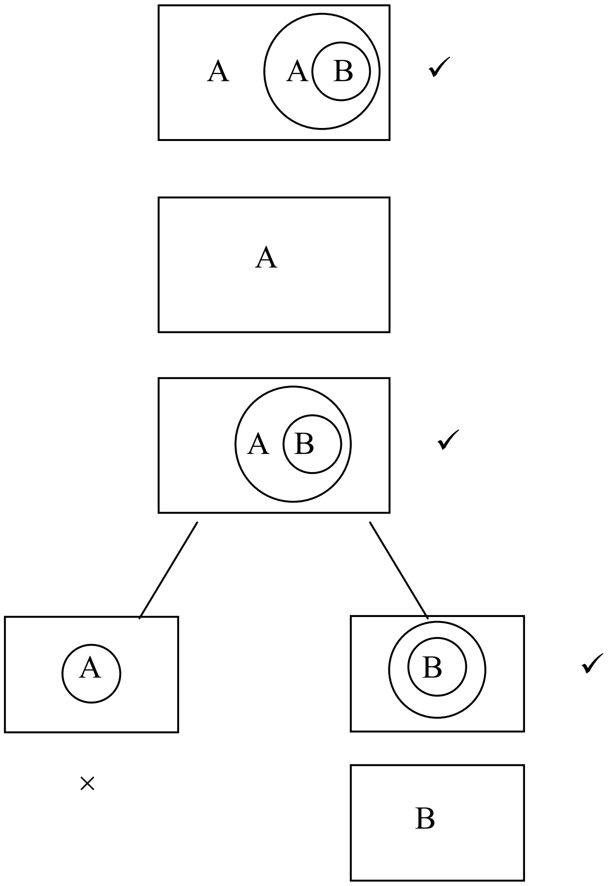
c) When applying DCD, let diagram D be written just above D'. Add a check mark next to D. For example:



d) When applying FDD just below diagram D, let diagram D be written just above D' and D" so that D' is juxtaposed to the left of D". Do *not* add a check mark next to D when applying this rule. Insert a line joining D and D' and insert a line joining D and D". For example:



e) If a branch closes (see 2e), place an '×' below the lowest diagram of that branch. For example:



2. Definitions. We shall need a rather involved set of definitions to discuss the nature of trees and prove the soundness of Peirce's five syntactic transformation rules for Alpha.

a) A diagram is decomposed if and only if it is either atomic or one of the decomposition principles has been applied to it.

b) A downward-exfoliating tree is a finite directed graph with a single diagram at the unique top node and with either zero, one, or two arrows emanating downward from each node and terminating at another single diagram. An upward-exfoliating tree is a finite directed graph in which the unique top node of a downward-exfoliating tree is relabelled 'bottom' and the arrows are said to exfoliate upward rather than downward.

c) A node is a diagram occurring in a tree.

d) A branch consists of all the diagrams that can be reached by starting with a diagram at the bottom of the tree and tracing a path up through the tree, ending with the diagram at the top node.

e) A closed branch contains at least two atomic diagrams, one of which has an atomic graph consisting of just one capital Roman letter and another of which has an atomic graph consisting of the same capital Roman letter with one cut around it. A branch that is not closed is open.

f) Completed open branch: an open branch such that every diagram on it is atomic or has been decomposed.

g) A completed tree is a tree each of whose branches is either closed or is a completed open branch. Let 'T_D', a decomposition tree for D, denote a completed tree for diagram D. (We will later show this is effectively unique.)

h) An open tree is a tree with at least one completed open branch.

i) We say that diagram D' occurs at level *i* in a tree if it can be reached by traversing *i* nodes connected by arrows beginning with the top node of the tree.

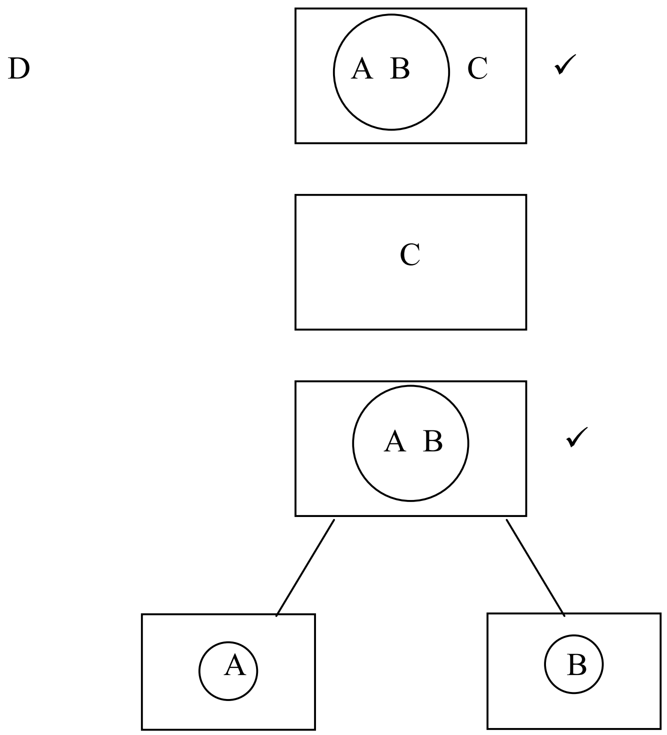
j) "path to a level of a tree": For any branch that extends to level *i* or further, that part of the branch that extends from the top (or bottom) node to level *i* is a *path* to level *i* and the path *contains* the set of diagrams that occur on it.

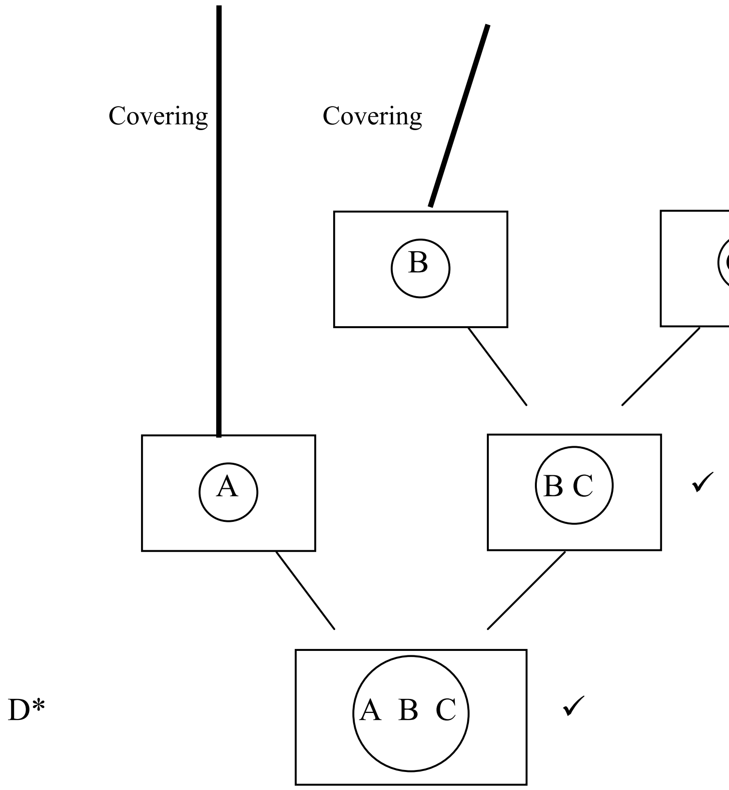
k) A branch of one complete tree covers a branch in another complete tree if and only if the set of atomic diagrams in the latter branch is a subset of the set of atomic diagrams in the former. One complete tree covers another if and only if every open branch in the former covers one or more open branches in the latter tree.

l) FDD-extension of T.' An FDD-extension of a completed tree T is a completed tree that results after zero or more applications of FDD to each open branch of T.

Remark. Since a tree is required to be a finite directed graph, the procedure of applying FDD and completing may only be done finitely many times.

m) Two completed trees form "coupled trees" if and only if one of the trees (the "top tree"), which exfoliates downward, covers the other tree (the "bottom tree"), which exfoliates upward. When speaking of "covering," we are referring to the branches of the top tree in relation to the branches of the bottom tree. Often, we shall use the decomposition tree for diagram D (T_D) to be the top tree and the decomposition tree for diagram D* (T_{D^*}) to be the bottom tree. For example, the following structure is formed by two coupled trees:





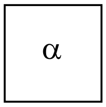
Here the top tree exfoliates downward from diagram D and the bottom tree exfoliates upward from diagram D*. I have added an additional notation device--a thick line--to prompt the reader to see the covering relationship between a branch of the top tree to a branch of the bottom tree. I shall follow this practice whenever coupled trees are displayed in the rest of this work.

n) A “tine” of D is either member of the set of diagrams D' and D" resulting from an application of DD upon diagram D. A “handle” of D is the set of diagrams resulting from applications of CD and/or DCD.

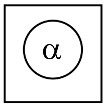
o) A fork of diagram D consists of a pair of tines of D resulting from a single application of DD to D or to a tine of D.

p) A CD-type diagram is a diagram to which CD may be applied. A DD-type diagram is a diagram to which DD may be applied. A DCD-type diagram is a diagram to which DCD may be applied.

q) Let a “singleton diagram of α” be the diagram consisting of the SA with only one graph, α, lying in the area of the SA. Let a “singleton diagram of cut-α” be the diagram consisting of the SA with only one graph, α, lying in the area of one cut, the cut lying in the area of the SA. We can illustrate schematically a singleton diagram of α as follows:



We can illustrate schematically a singleton diagram of cut-α as follows:



3. In order to determinately specify the structure of a tree we need the LPGN algorithm along with the following constraints.

Constraints for Decomposition Trees:

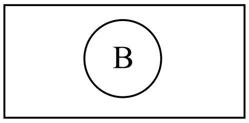
a) Decompose the topmost diagram in a tree.

b) Continue to decompose diagram(s) resulting from a previous decomposition until no further decompositions are possible.

4. Comment on FDD. We need FDD to address situations such as the following. Given the Semantic Entailment Definition (VIII.3), we would like to cover cases where from

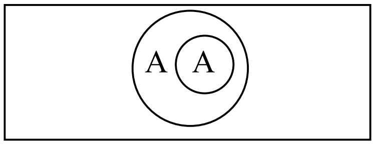
any statement we can entail a tautology. For example, we would like to show that statement B entails A or not-A. In diagrams, we want to be able to form a couple tree with diagram D,

D

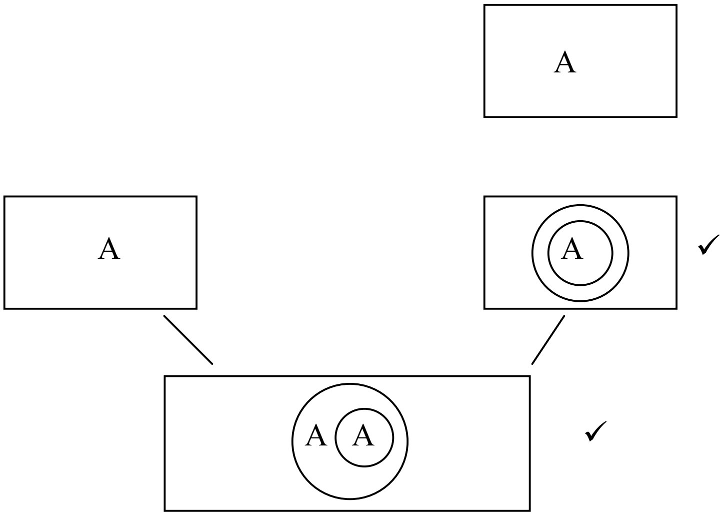


as the topmost node of a downward-exfoliating tree, and diagram D*,

D*

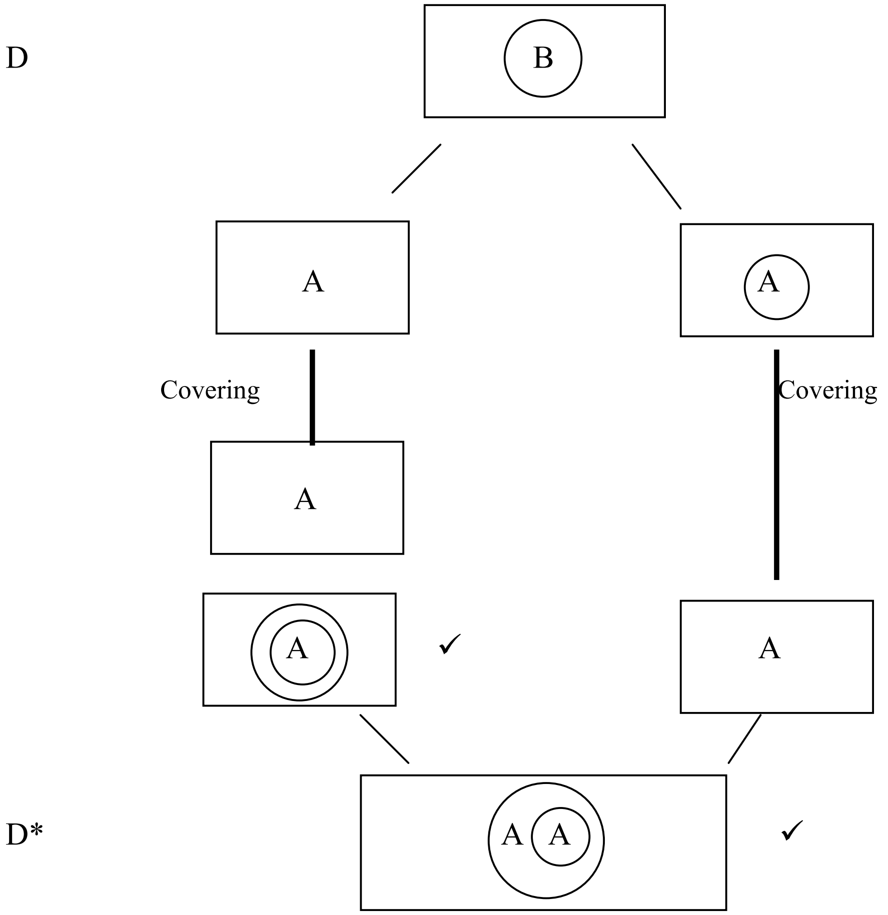


as the bottommost node of an upward-exfoliating tree. Since diagram D is atomic, we cannot apply CD, DD, or DCD to it. Thus T_D consists of just diagram D, and T_D is complete. Since diagram D* is not atomic—in fact it is a DD-type diagram—we can apply DD to it. We then could construct the following upward-exfoliating tree:



D*

The problem is that now all the branches of T_{D*} are complete, and thus T_{D*} is complete, but T_D does *not* cover T_{D*}. By applying FDD to the one open branch of T_D, we construct an extension of T_D (whose tree we call T) that *does* cover T_{D*}:



We shall use FDD in one place in the soundness proof (X.A.3.Case 2b).

VIII. Tree Theorems

In order to prove the soundness of the transformation rules specified in section IX, we need to ensure that the coupled trees are completed in a finite number of decomposition applications. In addition, we need to have the trees exfoliated in a single manner. Finally, we will require an explicit formalization of what will constitute semantic entailment within the context of coupled trees. We do so in 1-4 below.

Given a DD-type diagram D, we can specify exactly how many tines will result from applying DD to D on the basis of the number of graphs lying in an area of magnitude 1 in D; we can also locate which fork the tine of a singleton cut diagram will occur. In addition, given a CD-type diagram D, we can specify exactly the length of the handle resulting from applying CD to D on the basis of the number of graphs lying in an area of magnitude 0 in D. Furthermore, given a DCD-type diagram D, we can specify exactly the length of the handle resulting from applying DCD to D on the basis of the number of double cuts concatenated within the outermost double cut in D. We show all this in 5-8 below.

Finally, in the soundness proofs, we will need to refer to the fact that given a diagram D with a set of graphs lying in the same area of odd magnitude we are assured that in the decomposition tree of D we are guaranteed to have a DD-type diagram consisting of that same set of graphs lying in an area of magnitude 1. Similarly, we will need to refer to the fact that given a diagram D with a set of graphs lying in the same area of even magnitude we are assured that in the decomposition tree of D we are guaranteed to have a CD-type diagram consisting of that same set of graphs lying in an area of magnitude 0. We show these facts in 9-10 below.

1. **Completed Tree Theorem**: For any diagram D, a completed tree can be constructed.

Proof: By the Diagram Reduction Theorem (VI.8), each application of a decomposition principle to a given diagram shall result in diagram(s) that contain fewer tokens than in the given diagram. Since any given diagram has a finite number of tokens, repeated applications of the decomposition principles will eventually result in all diagrams being either atomic or decomposed. But then the corresponding tree will consist of branches each of which is either closed or a completed open branch, and so the corresponding tree will be complete.

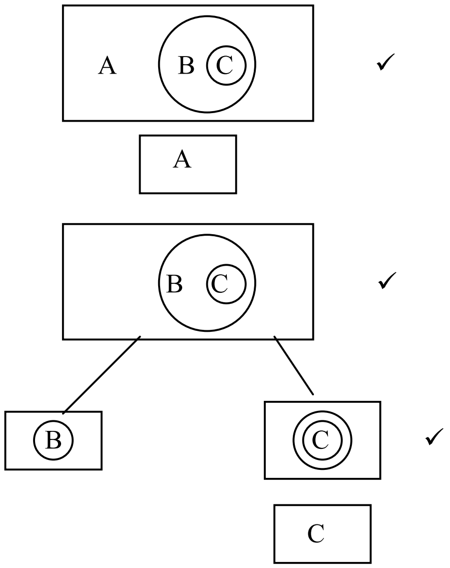
2. **Unique Completed Tree Theorem**: For any diagram D, there is a unique completed decomposition tree for that diagram.

Proof: At each stage, which decomposition rule is to be applied is dictated effectively by the LPGN constraint and the fact that it is only possible to apply one of the tree decomposition rules at any step, so the decomposition tree is unique.

3. **Semantic Entailment Definition (SED)**: D |= D* if and only if ∃ tree T which is an FDD-extension of T_D such that T covers T_{D*}.

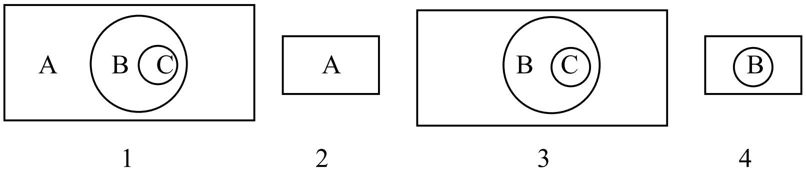
Illustration 1: Consider the completed tree for the following diagram D:

D

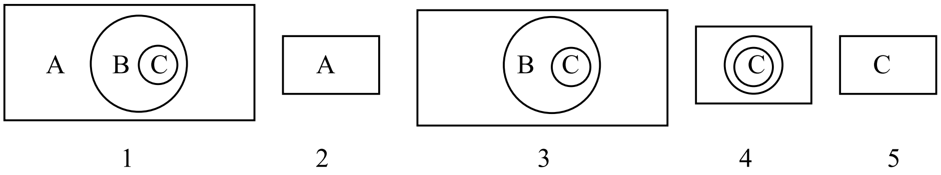


Observe the two open paths of this tree:

(i)

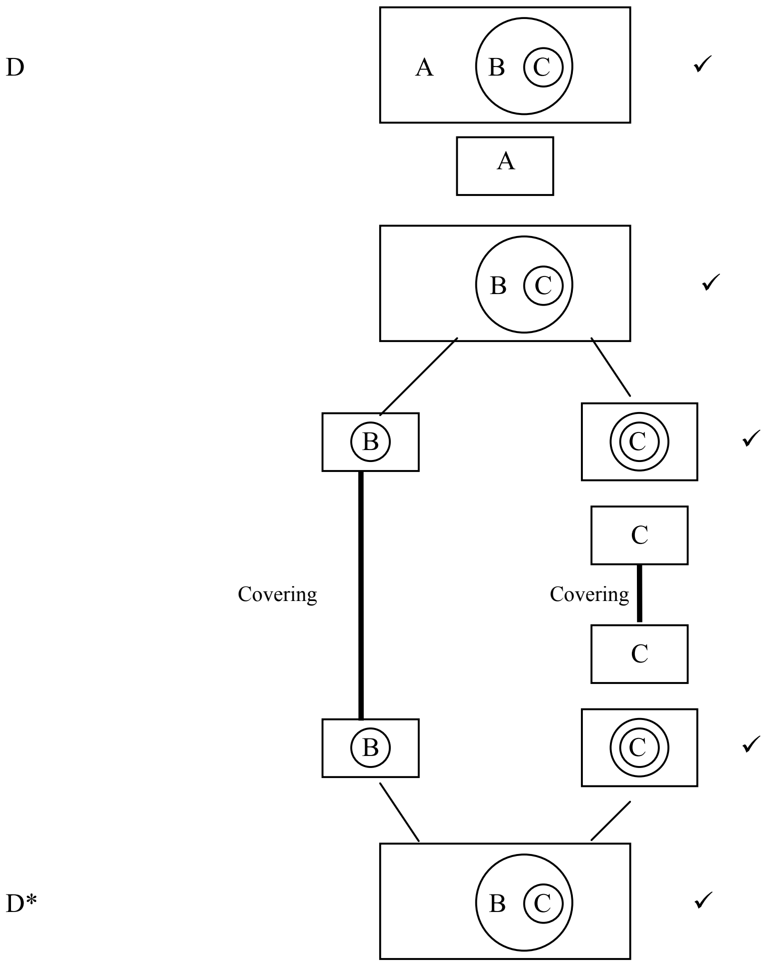


(ii)

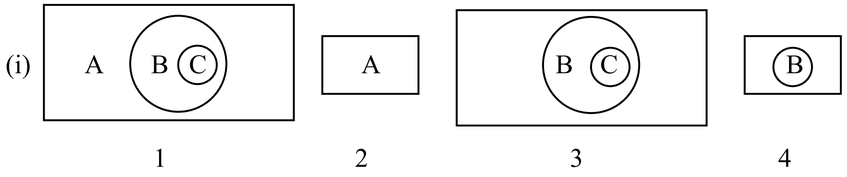


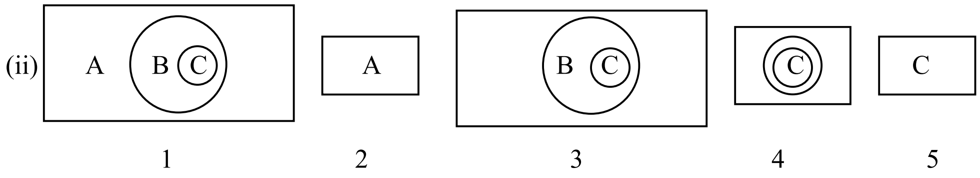
From path (i) we can pick out the atomic diagrams (2, 4), that is, the set of diagrams that contain the basic semantic commitment of that path, viz., that A is true and B false. Similarly, from path (ii) we can pick out the atomic diagrams (2, 5) that represent the basic semantic commitment of that path, viz., that A is true and C is true.

Now consider the following coupled tree structure between diagrams D and D*, where T_D covers T_{D*}:

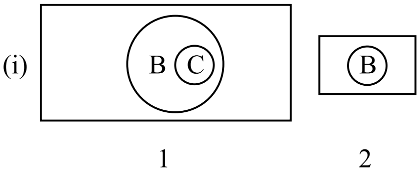


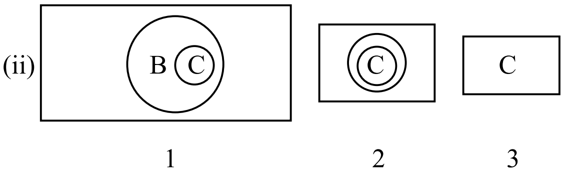
Observe the two open paths of T_D:





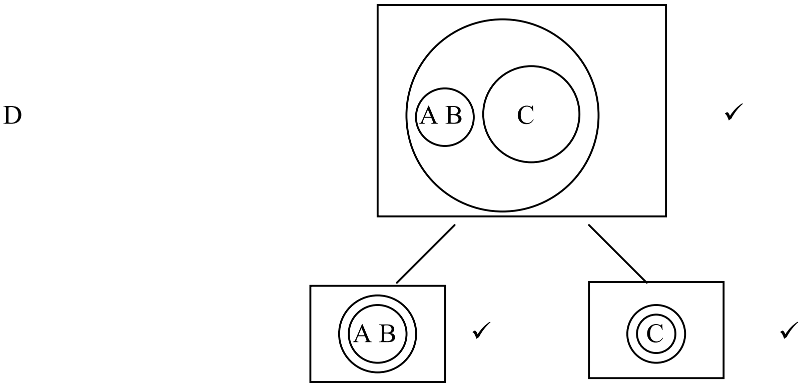
and of T_{D*}:

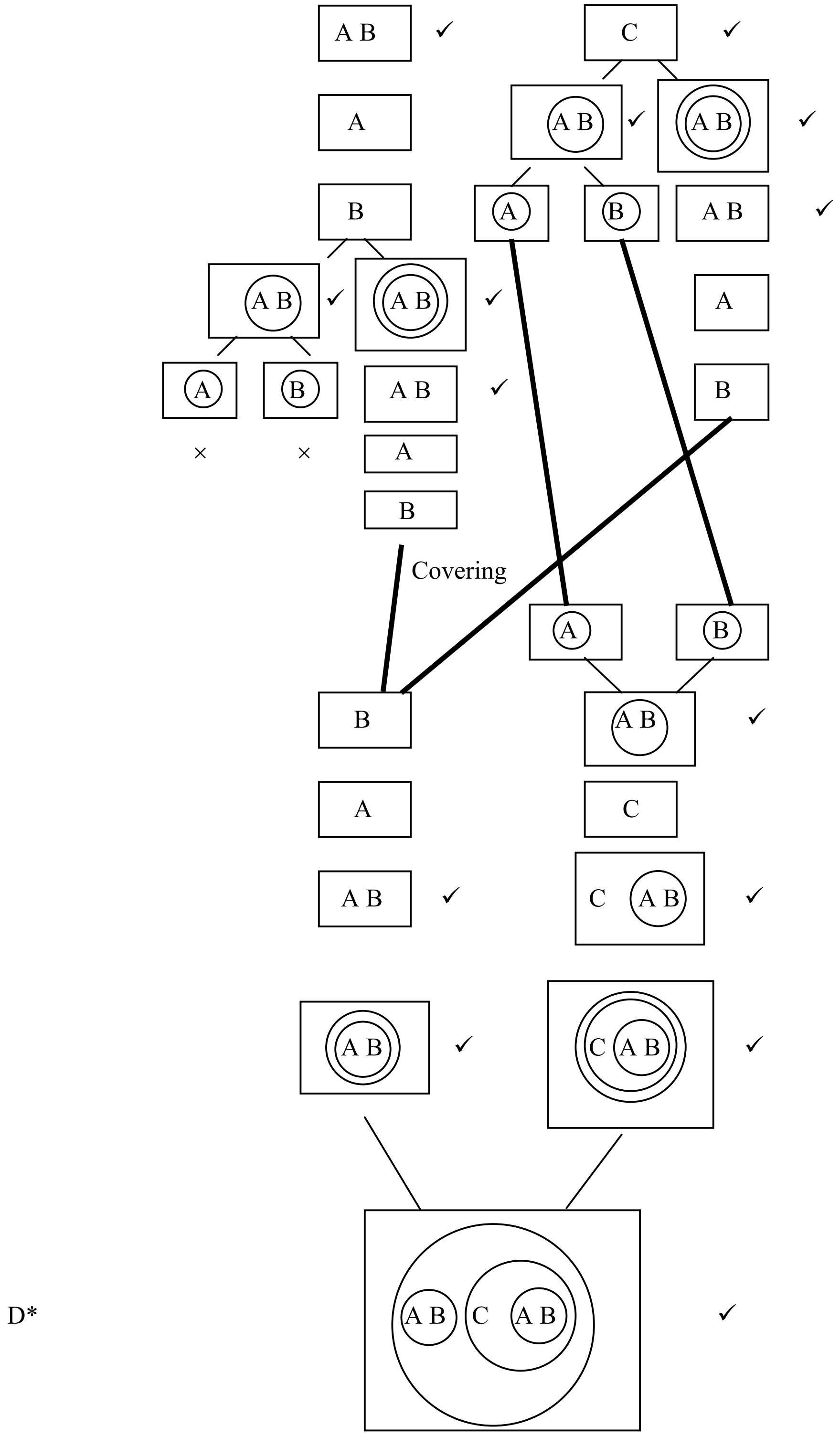




For the set of diagrams representing the basic semantic commitments of path (i) of T_D (2, 4), there is a path of T_{D*}, viz., path (i), that is no stronger in terms of basic semantic commitment than path (i) of T_D. Otherwise put, the set of atomic diagrams in path (i) of T_{D*} (2) is a subset of the set of atomic diagrams in path (i) of T_D (2, 4). Likewise, the set of atomic diagrams in path (ii) of T_{D*} (3) is a subset of the set of atomic diagrams in path (ii) of T_D (2, 5).

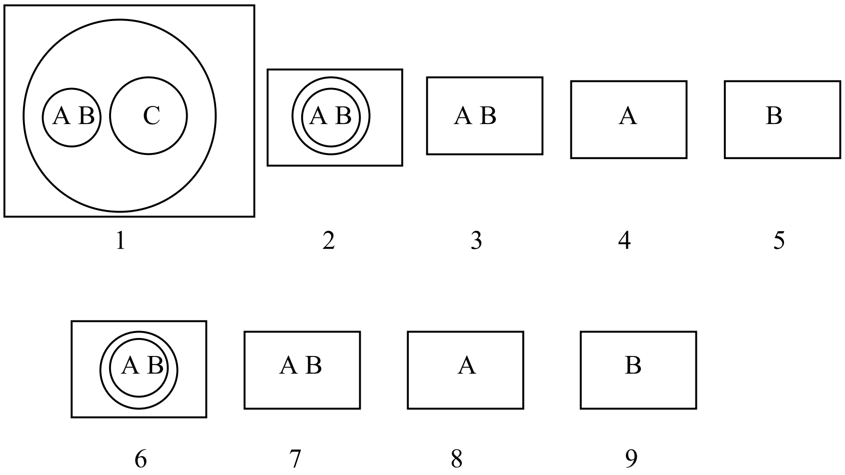
Illustration 2: Consider the following coupled tree structure between diagrams D and D*, where T (an FDD-extension of T_D) covers T_{D*}:



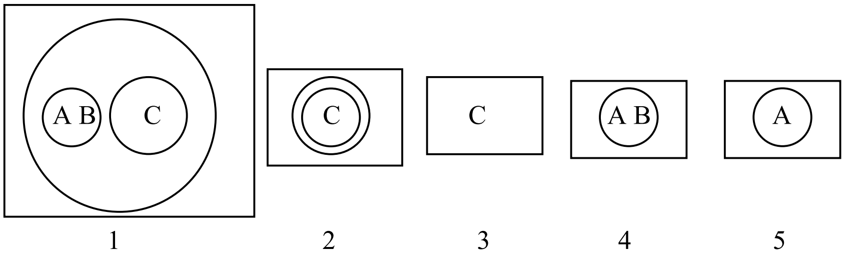

A B ✓
C ✓
A
A B ✓
A B ✓
B
A
B
A B ✓
A B ✓
A B ✓
A
A
B
A B ✓
B
×
×
A
B
Covering
A
B
B
A B ✓
A
C
A B ✓
C A B ✓
A B ✓
C A B ✓
D*
A B C A B ✓

Observe the four open paths of T:

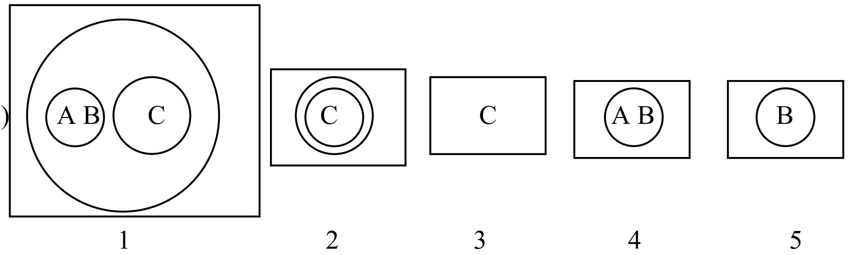
(i)



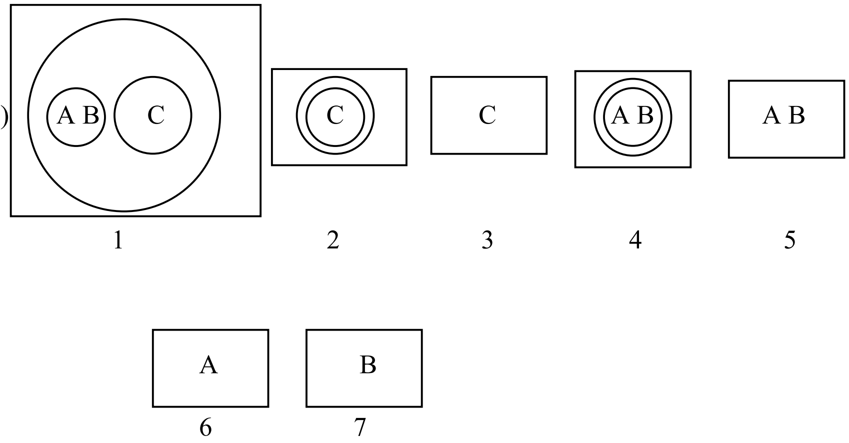
(ii)



(iii)

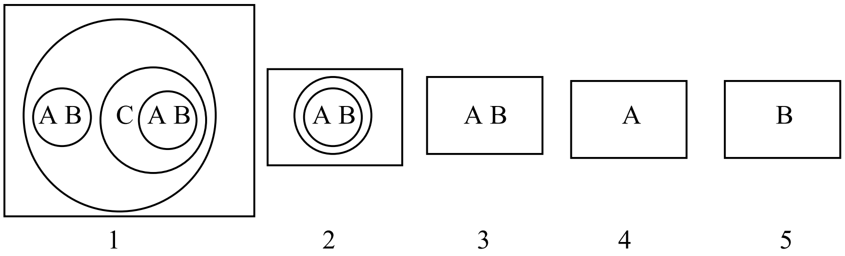


(iv)

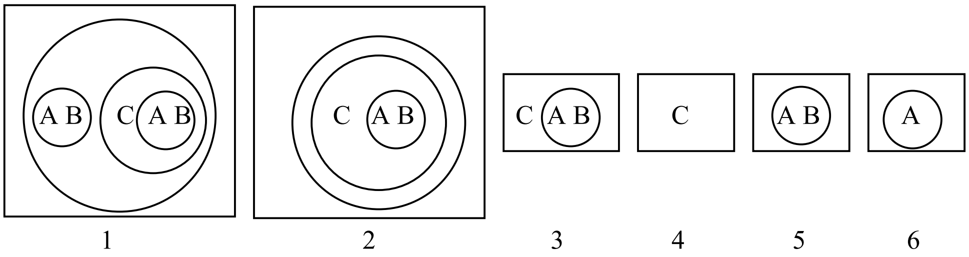


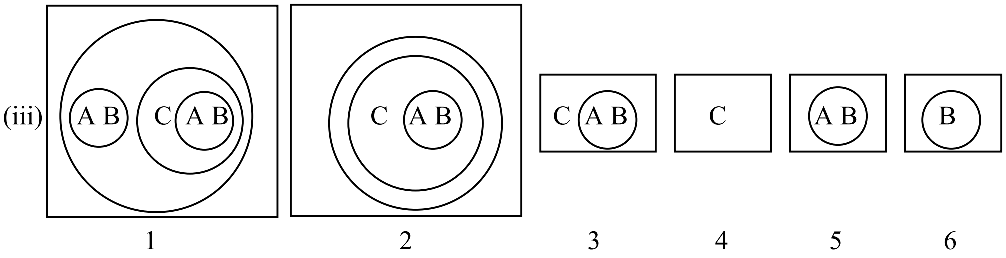
and the three open paths of T_{D*}:

(i)



(ii)





For the set of diagrams representing the basic semantic commitments of path (i) of T_D (4, 5), there is a path of T_{D*}, viz., path (i), that is no stronger in terms of basic semantic commitment than path (i) of T_D. Otherwise put, the set of atomic diagrams in path (i) of T_{D*} (4, 5) is a subset of the set of atomic diagrams in path (i) of T_D (4, 5). In addition, the set of atomic diagrams in path (i) of T_{D*} (4, 5) is a subset of the set of atomic diagrams in path (iv) of T_D (3, 6, 7). Likewise, the set of atomic diagrams in path (ii) of T_{D*} (4, 6) is a subset of the set of atomic diagrams in path (ii) of T_D (3, 5). Lastly, the set of atomic diagrams in path (iii) of T_{D*} (4, 6) is a subset of the set of atomic diagrams in path (iii) of T_D (3, 5).

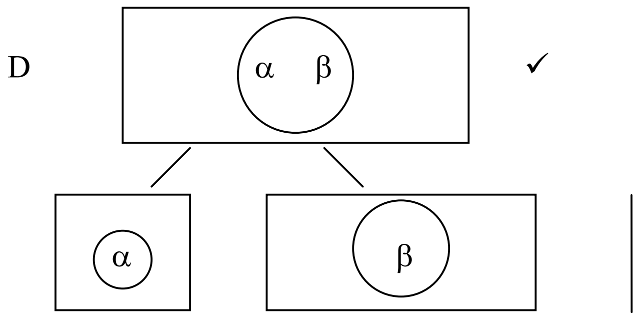
4. **Principle of Semantic Transitivity**: If D |= D* and D* |= D**, then D |= D**

Proof: Assume D |= D* and D* |= D**. Then by SED there exists a tree T which is an FDD-extension of T_D such that T covers T_{D*} and there exists tree T' which is an FDD-extension of T_{D*} such that T' covers T_{D**}. Since T covers T_{D*}, for each open branch of T, there exists an open branch of T_{D*} such that the atomic information in the latter branch is a subset of the atomic information in the former branch. Similarly, since T' covers T_{D**}, for each open branch of T' there exists an open branch of T_{D**} such that the atomic information in the latter branch is a subset of the atomic information in the former branch. Let T''' be the FDD-extension of tree T such that every application of FDD to T_{D^*} is also applied to T. Then for each open branch of T''' there exists an open branch of $T_{D^{**}}$ such that the atomic information in the latter branch is a subset of the atomic information in the former branch. But T''' is an FDD-extension of tree T and hence an FDD-extension of T_D. So, there exists a tree that is an FDD-extension of T_D such that T covers $T_{D^{**}}$. Therefore, D |= D**.

5. **Tine numbering theorem for DD-type diagrams**: Let α, β, γ, . . . κ be a set of *n* graphs lying in an area of magnitude 1 in a DD-type diagram D, where α, β, γ, . . . κ are graphs not further subject to DD. Then the number of tines of D = 2(*n*-1) for *n* ≥ 2.

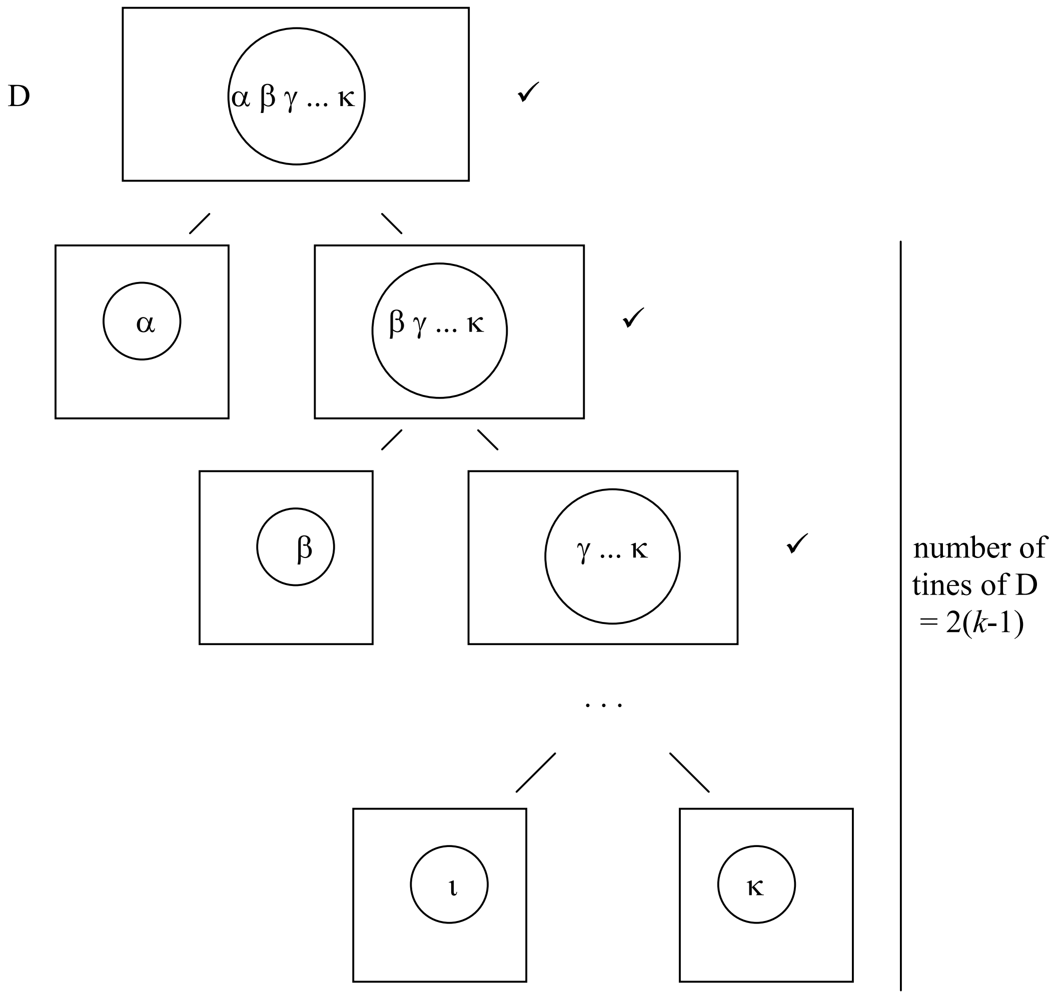
Proof:

Basis case. *n* = 2. Then the number of tines of D = 2(2-1) = 2:

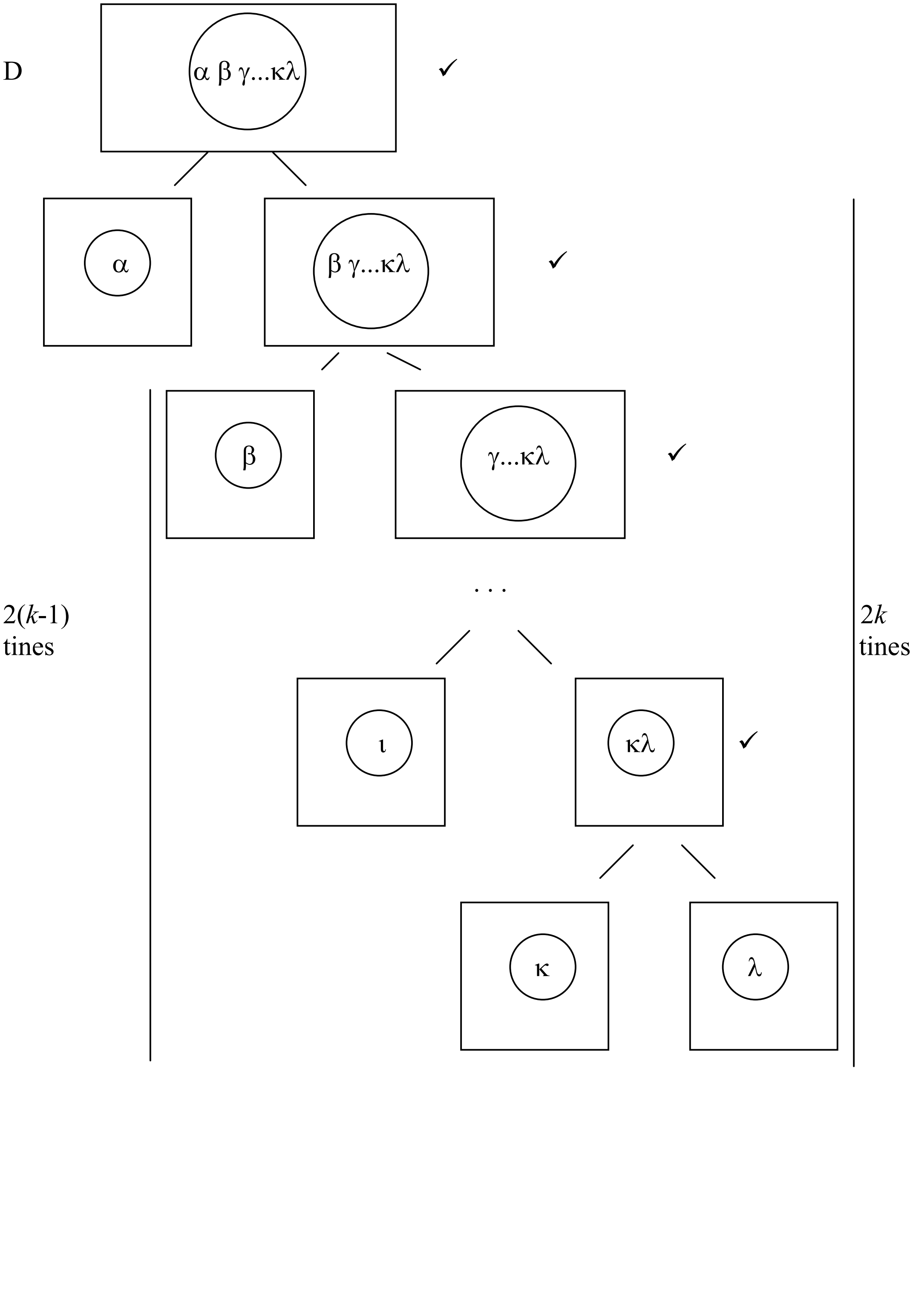


number of tines of D = 2

Inductive step. Assume that if *n* = *k*, then the number of tines of D = 2(*k*-1):



If n = k+1, then the number of tines of D = 2([k+1]-1) = 2k:



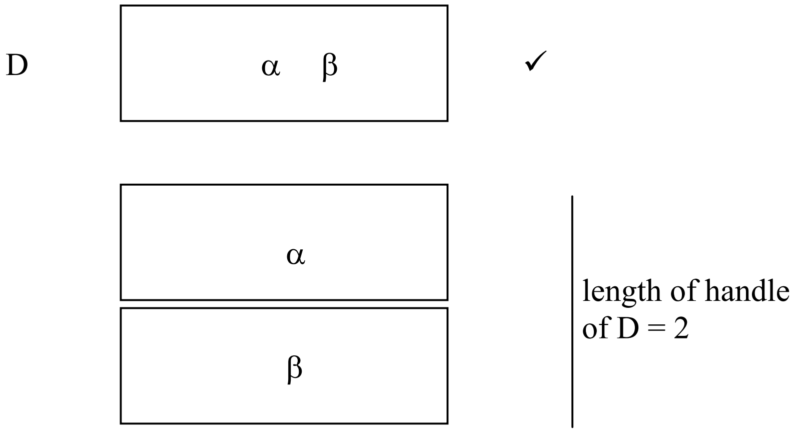
Note that we first apply the rule DD once at the top of this tree structure and then apply the inductive hypothesis to that part of the tree structure that is bounded by the vertical line on the left.

■

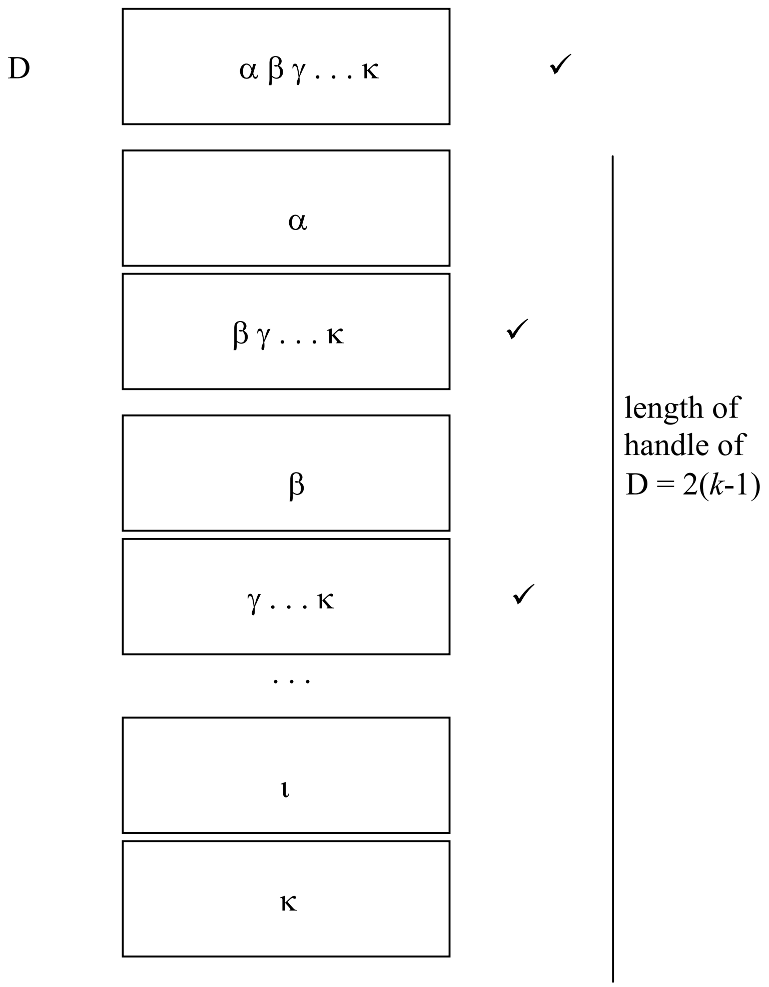
6. **Handle length numbering theorem for CD-type diagrams**: Let α, β, γ, . . . κ be a set of *n* graphs lying in an area of magnitude 0 in a CD-type diagram D, where α, β, γ, . . . κ are graphs not further subject to CD. Then the length of the handle of D = 2(*n*-1) for *n* ≥ 2.

Proof:

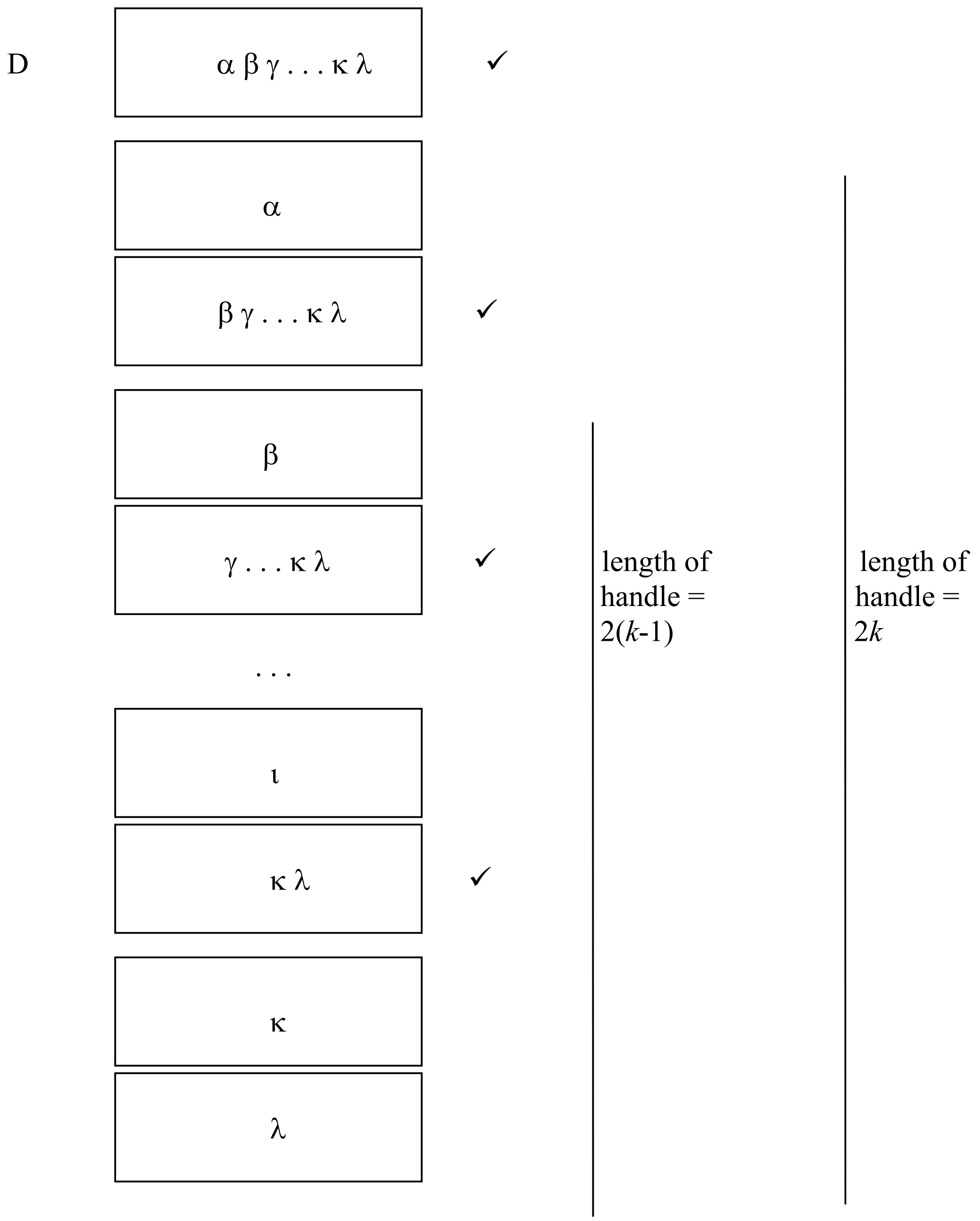
Basis case. *n* = 2. Then the length of the handle of D = 2(2-1) = 2:



Inductive step. Assume that $n = k$, then the length of the handle of D = 2(k-1):



If $n = k+1$, then the length of the handle of D = 2([k+1]-1) = 2k:



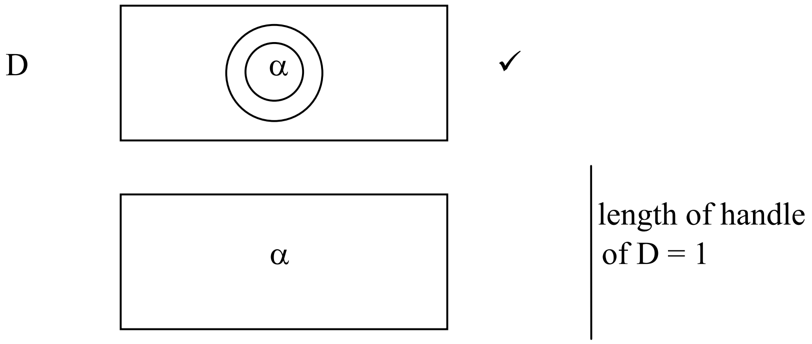
Note that we first apply the rule CD once at the top of this tree structure and then apply the inductive hypothesis to that part of the tree structure that is bounded by the leftmost vertical line.

■

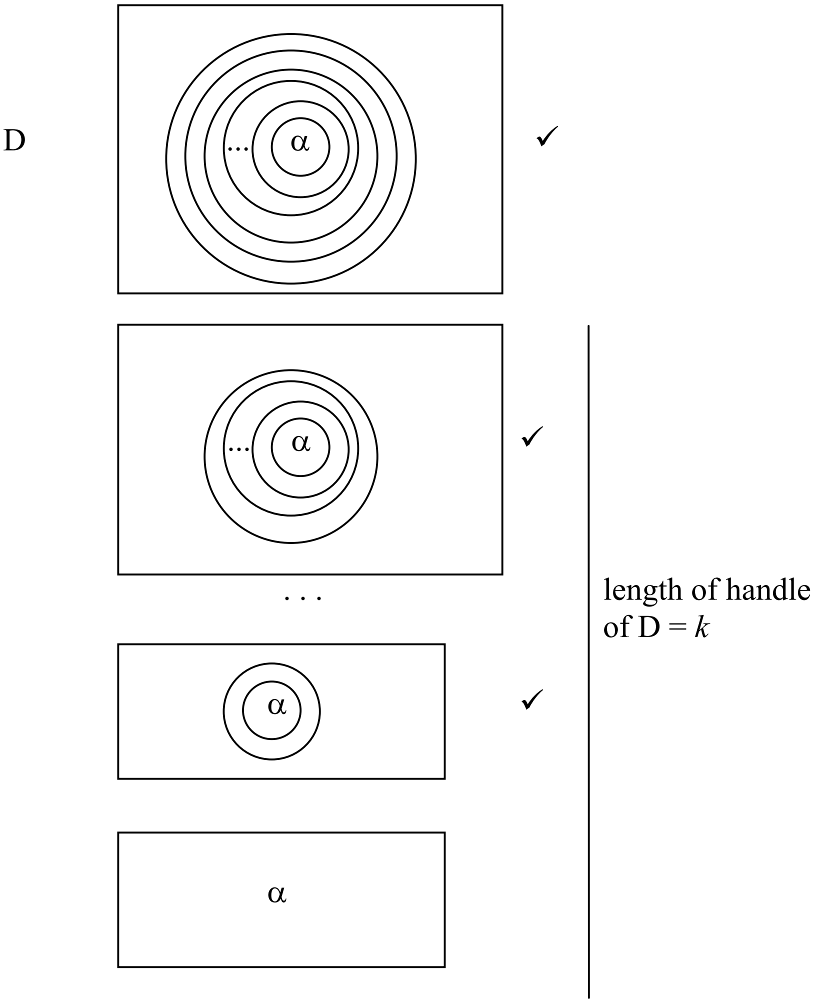
7. **Handle length numbering theorem for DCD-type diagrams**: Let α be a graph lying in an area of magnitude 2*n* in a DCD-type diagram D with *n* double cuts around α and α is not further subject to DCD. Then the length of the handle of D = *n* for *n* ≥ 1.

Proof:

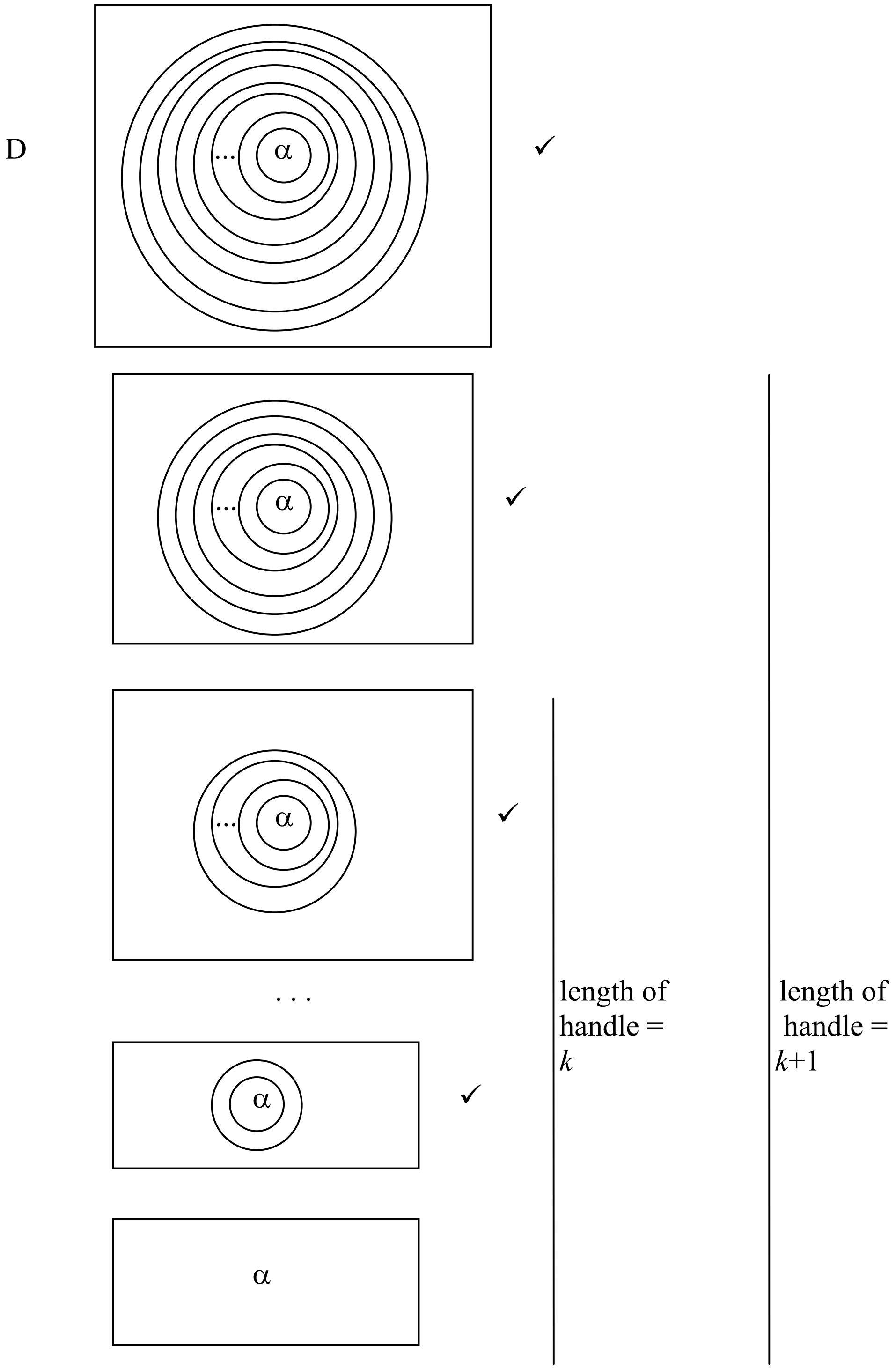
Basis case. *n* = 1. Then the length of the handle of D = 1:



Inductive step. Assume that if *n* = *k*, then the length of the handle of D = *k*:



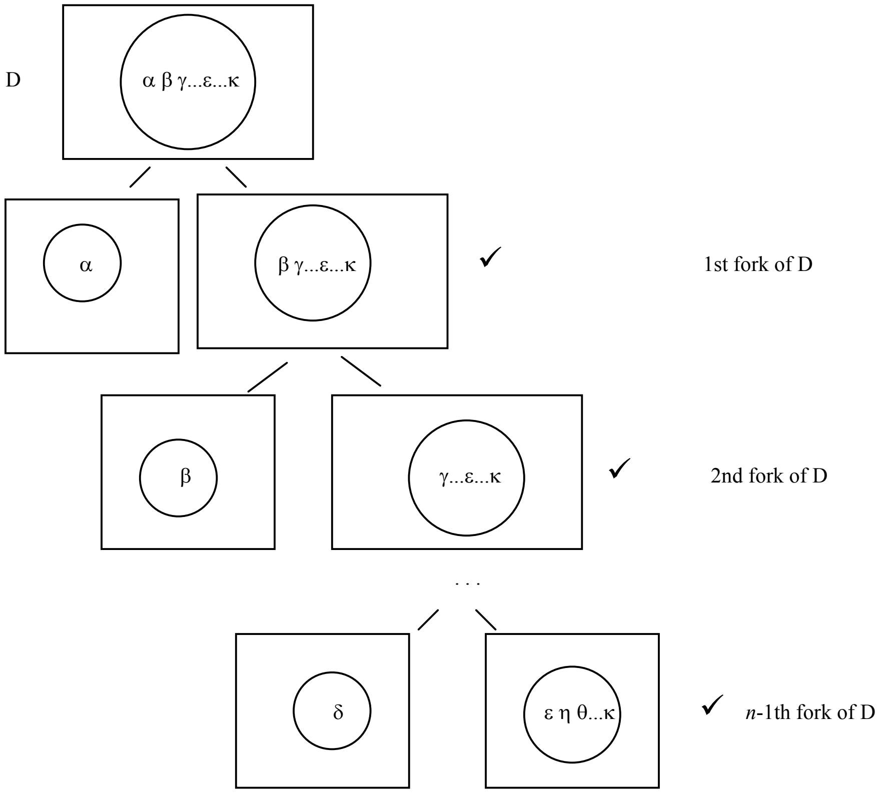
If *n* = *k*+1, then the length of the handle of D = *k*+1:

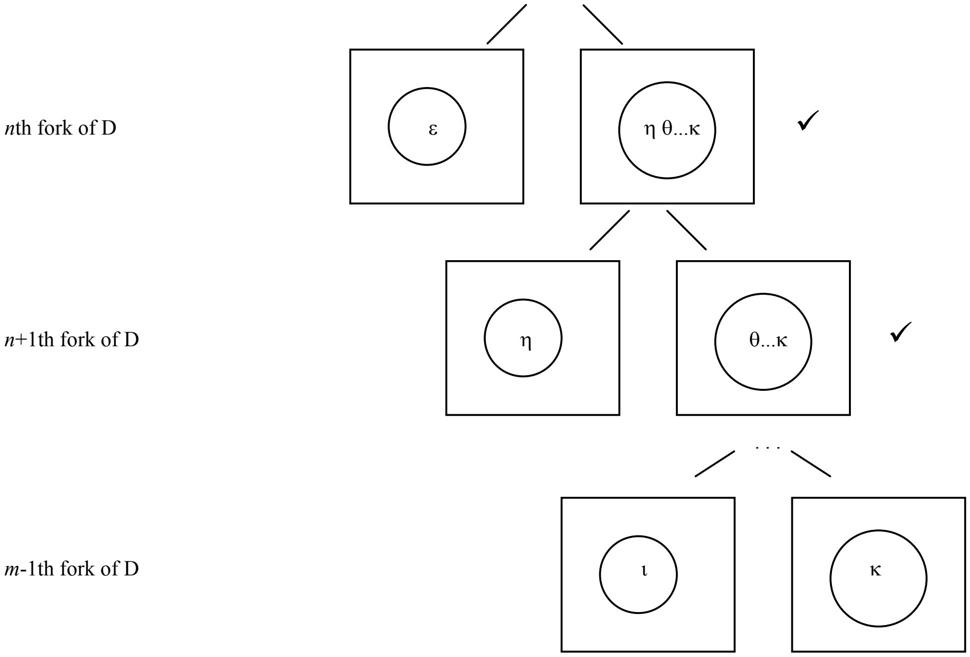


Note that we first apply the rule DCD once at the top of this tree structure and then apply the inductive hypothesis to that part of the tree structure that is bounded by the leftmost vertical line.

■

8. **Fork numbering theorem**: Let ε be a graph in the set of *m* graphs α, β, γ, . . . κ lying in an area of magnitude 1 in a DD-type diagram D, where α, β, γ, . . . κ are graphs not further subject to DD, and let ε be the *n*th graph to receive a tine of the singleton diagram of cut-ε. Then the tine of the singleton diagram of cut-ε will occur on the *n*th fork of D for $n \geq 1$. Illustration:

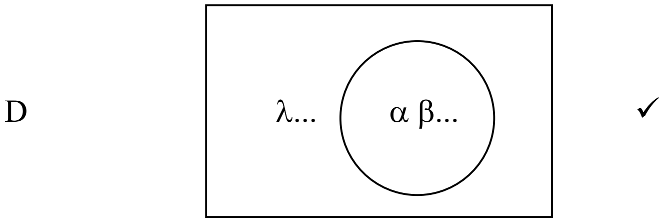


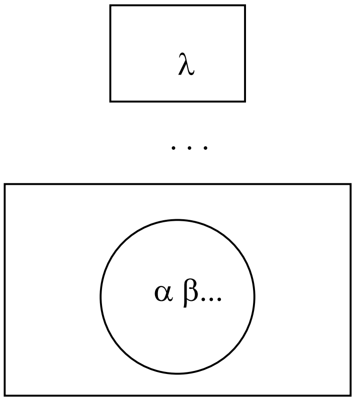


9. **Area reduction theorem for areas of odd magnitude**: Let set A = {α, β, . . .} of two or more graphs be all the graphs lying in the same area of odd magnitude 2*n*-1 in diagram D. Then in T_D there exists at least one DD-type diagram consisting of the SA, just one cut lying in the area of the SA, and just the graphs in set A lying in the area of the cut.

Proof:

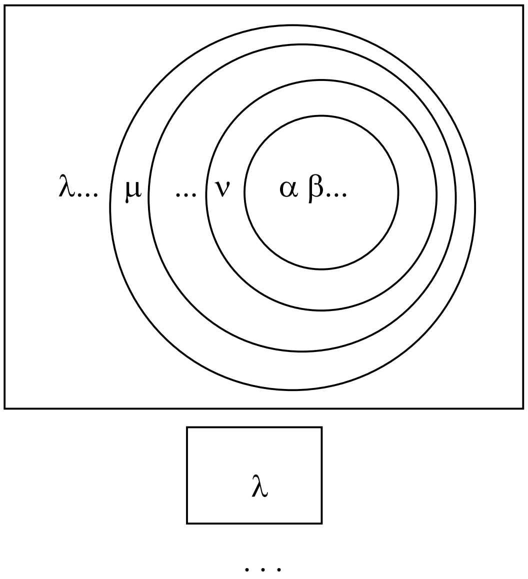
Basis case. *n* = 1, the area magnitude = 2(1)-1 = 1.



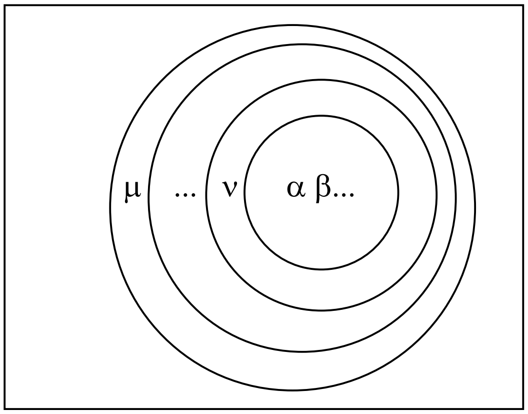


Inductive step. *n* = *k*, the area magnitude = 2*k*-1.

D

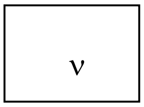


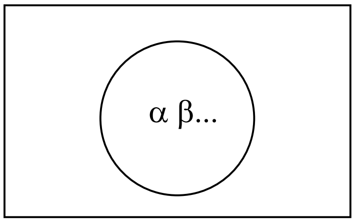
✓



✓

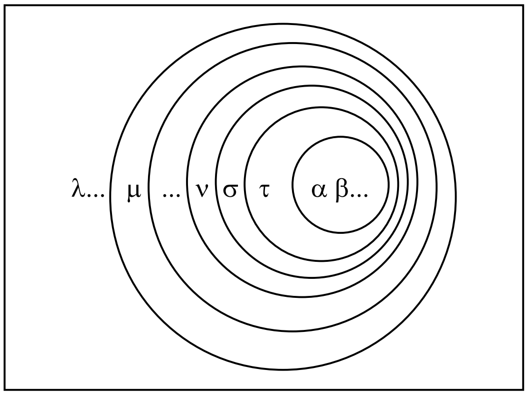
. . .



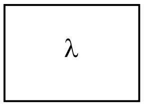


n = *k*+1, the area magnitude = 2*k*+1.

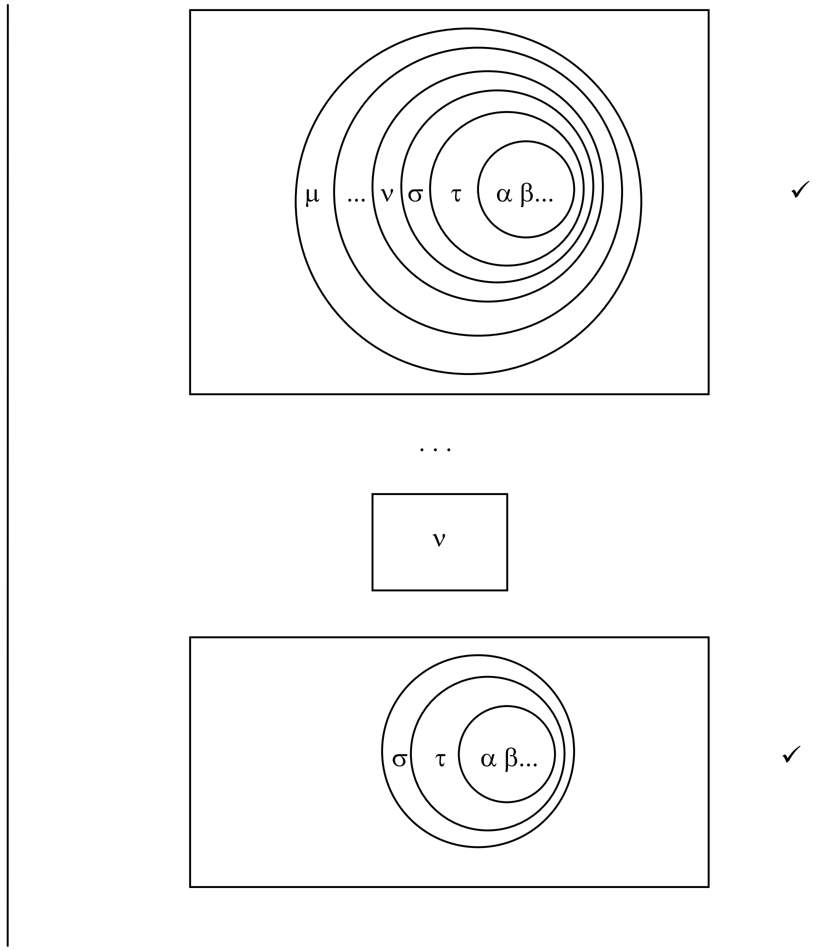
D

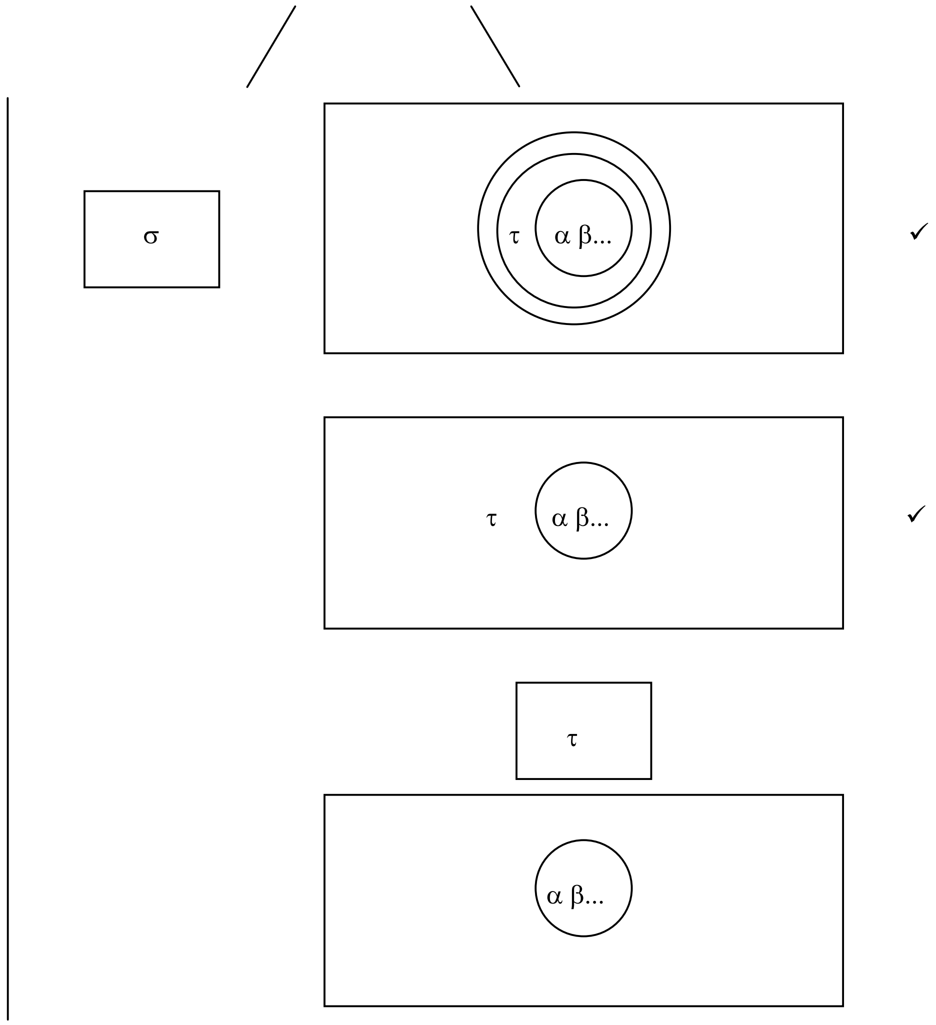


✓



. . .


μ ... ν σ τ α β...
✓
. . .
ν
σ τ α β...
✓



Note that we first apply the rule CD once at the top of this tree structure and then apply the inductive hypothesis to that part of the tree structure that is bounded by the vertical line.

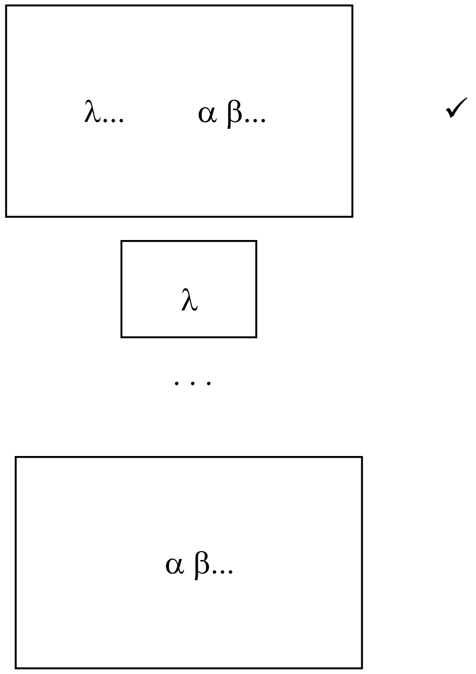
■

10. **Area reduction theorem for areas of even magnitude**: Let set A = {α, β, . . .} of two or more graphs be all the graphs lying in the same area of even magnitude 2*n*-2 in diagram D. Then in T_D there exists as least one CD-type diagram consisting of the SA and just the graphs in set A lying in the area of the SA.

Proof:

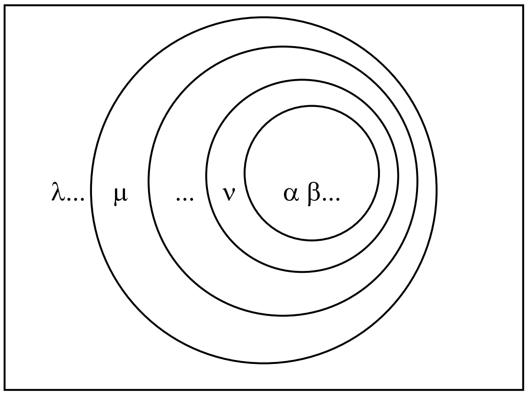
Basis case. *n* = 1, the area magnitude = 2(1)-2 = 0.

D

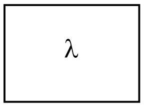


Inductive step. *n* = *k*, the area magnitude = 2*k*-2.

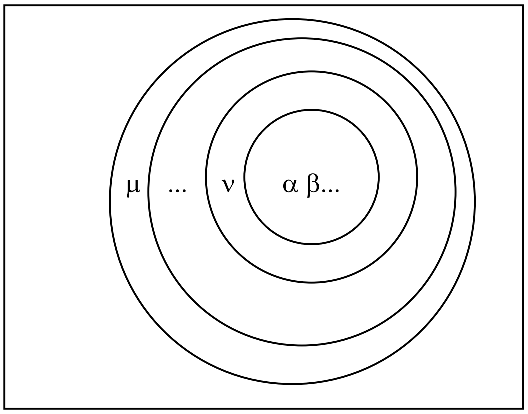
D



✓

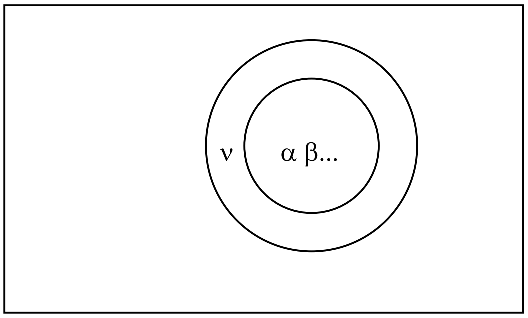


. . .

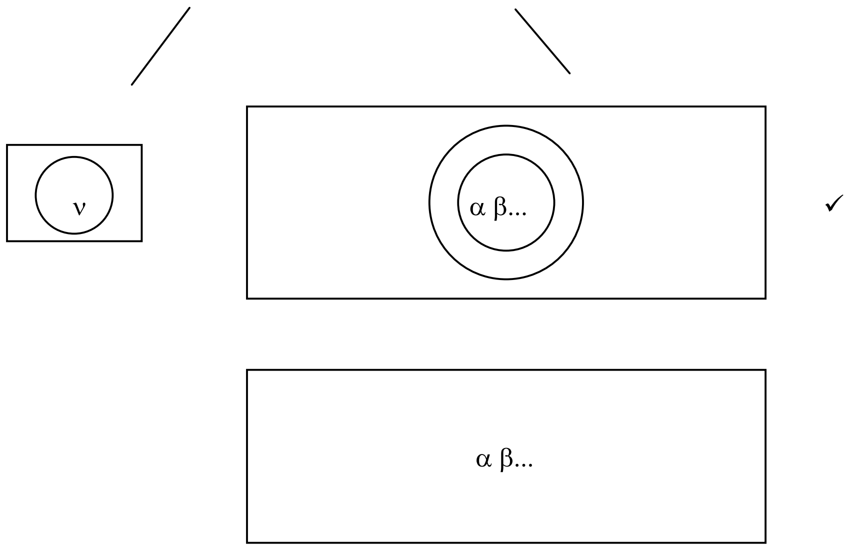


✓

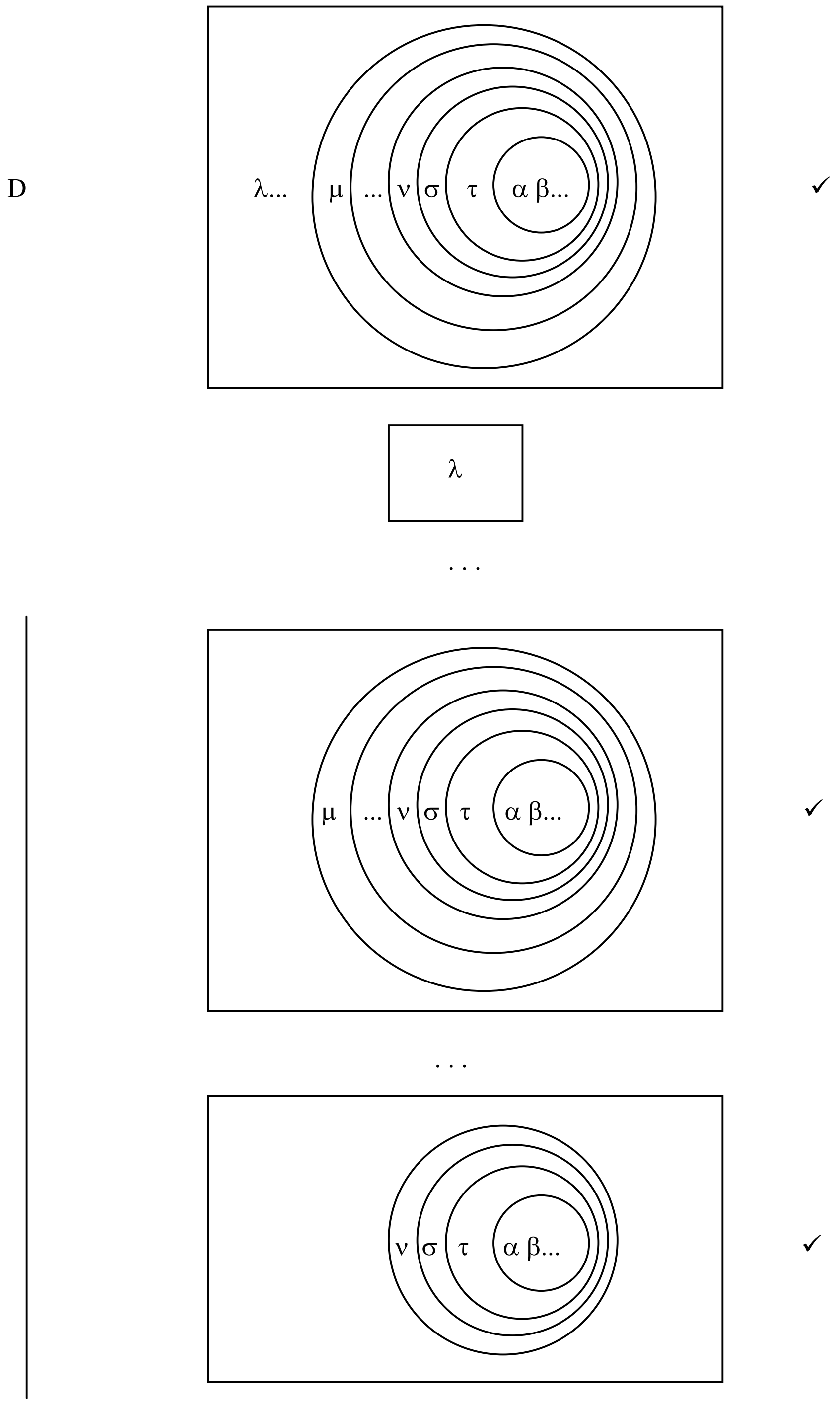
. . .

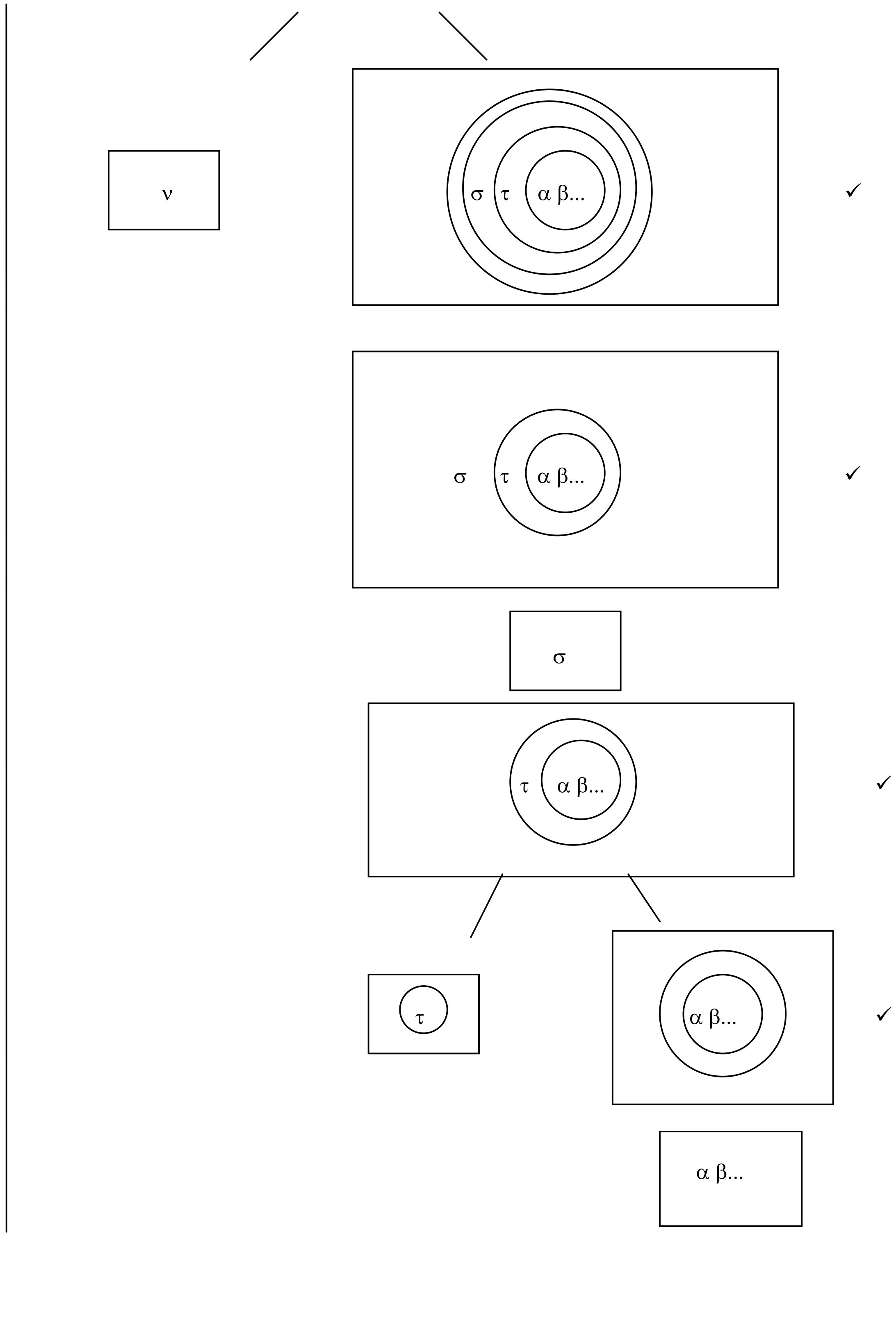


✓


ν
α β…
✓
α β…

$n = k+1$, the area magnitude = $2k$.




ν
σ τ α β...
✓
σ τ α β...
✓
σ
τ α β...
✓
τ
α β...
✓
α β...

Note that we first apply the rule CD once at the top of this tree structure and then apply the inductive hypothesis to that part of the tree structure that is bounded by the vertical line.

■

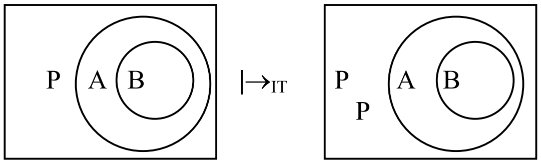
IX. Syntactic Transformation Rules

These formulations of the transformation rules are as found in Roberts, 1973, pp. 41-44.

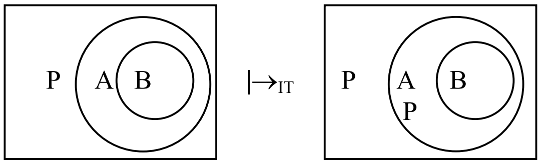
1. Insertion (IN): Any graph may be scribed on any oddly enclosed area.
2. Erasure (E): Any evenly enclosed graph may be erased.
3. Iteration (IT): If a graph P occurs on SA or in a nest of cuts, it may be scribed on any area not part of P, which is contained by *[[*P*]]*. (An extended discussion of this rule follows below.)
4. Deiteration (DE): Any graph whose occurrence could be the result of iteration may be erased.
5. Double Cut (DC): The double cut may be inserted around or removed (where it occurs) from any graph on any area.

In the soundness proofs, we should note the following. When performing Insertion, we may place any well-formed graph on an area of magnitude $2n$-1 in diagram D. When performing Deiteration, for two tokens of graph α such that one token lies on an area of magnitude m within a nest of cuts in diagram D and the second token lies on an area of magnitude $\geq m$ within that same nest of cuts, we may erase the second token.

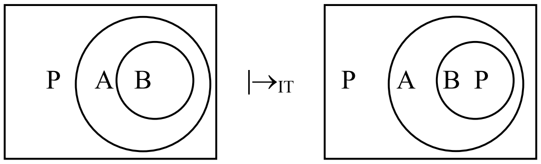
The formulation of the Iteration rules deserves an extended comment. As we recall from III.8, the place of a cut or graph is the area on which a cut is made or a graph is scribed. We are letting the place be symbolized by ‘*[[*P*]]*’. In doing so, we are departing from other formulations of this rule (e.g., Roberts, 1973, Shin) that use ‘{P}’ instead of our ‘*[[*P*]]*’. We do this so that we do not confuse the place of a cut or graph with a set of cuts or graphs. Now let us consider some sets of inferences using the Iteration rule to see why we have adopted the formulation of the rule as we have. As we shall note at the beginning of section X, we use the symbol ‘|→’ to denote a single application of a transformation rule to derive one diagram from a second. The first set of diagrams show an iteration of P in the same area as the original P subject to the non-overlapping formation rule (II.3.a.ii):



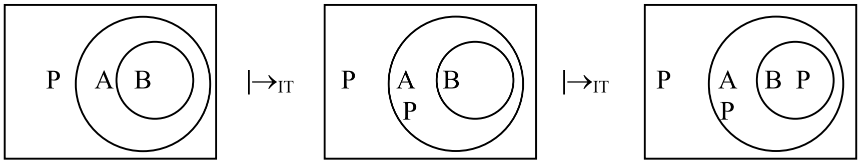
The next set of diagrams show the iteration of P upon a nest of cuts contained by the area of the original instance of P:



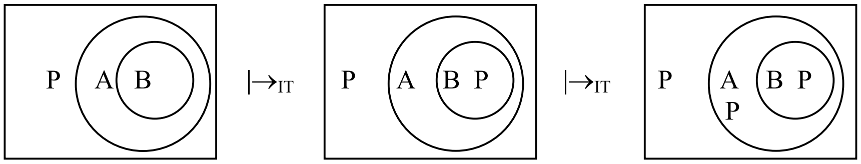
And similarly, the next set of diagrams show the iteration of P upon a nest of cuts contained by the area of the original instance of P, although in this case the iteration of P is placed into an area of magnitude 2 in the nest:



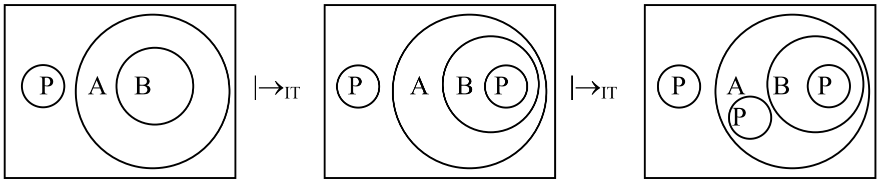
Now in the following set of three diagrams, we witness the progressive iteration of P first into an area of magnitude 1 of the nest, and then into an area of magnitude 2 of the nest:



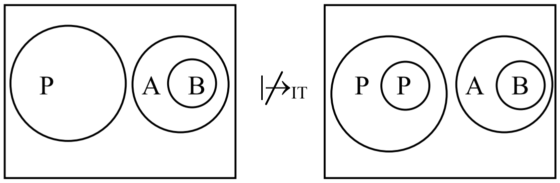
One interesting feature of the above set of transformations is that we could regard the third diagram as derived from second diagram on the basis of the iteration of either the instance of P on the SA or the instance of P in the area of magnitude 1. The following set of diagrams does not have a similar choice of which P is to be iterated to derive the third diagram:



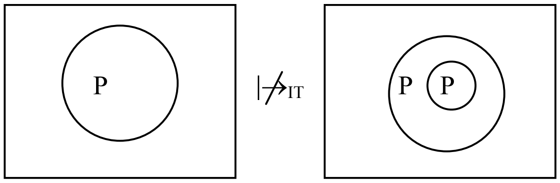
The instance of P in an area of magnitude 1 in the third diagram could *not* have come about due to the instance of P in an area of magnitude 2 in the second diagram, for the area of magnitude 1 in the nest is not contained by the area of magnitude 2. However, the nest's area of magnitude 1 is contained by the area of the SA, upon which graph P occurs. Thus, it is *only* the iteration of P resting in the SA that warrants the inference from the second to third diagram. The following set of diagrams show an analogous set of transformations to the above, except that instead of the iterated graph being P, it is cut-P:



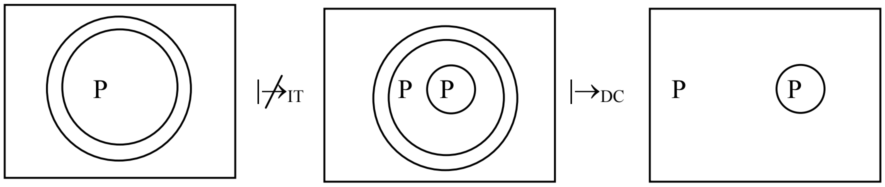
In the following set of diagrams we see the importance of the part of the rule that stipulates that the iterated graph “may be scribed on any area not part of P.” In the previous transformations, simply obeying the non-overlapping formation rule (II.3.a) would have been enough. However, that is not enough for all cases of iteration, as we see in the following invalid transformation:



Here, the original graph cut-P has been iterated into an area of itself. (I am using ‘$\not\mapsto_{IT}$’ to denote an illicit use of Iteration.) What is the problem with this? Well in one case, we would get what appears to be an innocent transformation:



This is an "innocent" transformation in that the second diagram represents a tautology. (We do not need this illicit use of Iteration to derive this tautology: we could use the Insertion rule.) However, as we might expect, if we enclose the entire set of graphs in the above diagrams within one cut, we will derive a contradiction:

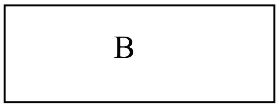


It is somewhat disturbing that Don Roberts, in a later work (Roberts, 1997, p. 391), formulates the Iteration rule as follows: "A graph which already occurs may be scribed again within the same or additional cuts." This formulation, though more concise than we one we adopt in this work, seems to leave open the illicit use of Iteration we have just identified.

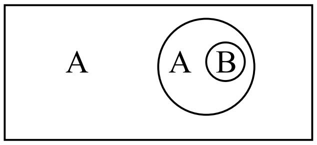
An important point to note regarding the derivation of diagrams is the following. Peirce envisioned the applications of these rules upon a set of graphs in a single diagram,

> Our purpose, then, is to study the workings of necessary inference. What we want, in order to do this, is a method of representing diagrammatically any possible set of premises, this diagram to be such that we can observe the transformation of these premises into the conclusion by a series of steps each of the utmost possible simplicity. (4.429)

Consider the example of *modus ponens*. We would want to derive the diagram,



from

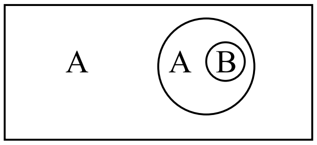


As Roberts notes,

> There are in fact two ways to do this. One way is to perform the transformations on the premisses themselves, so that when the inference is completed only the conclusion remains on SA. The other, and more instructive way, is to perform the actual transformations on iterated instances of the premisses, so that when the inference is completed the steps leading to the conclusion remain on SA as a kind of record. (1973, p. 45)

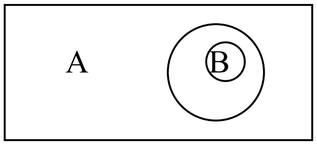
As is common with other writers on the Existential Graphs, e.g., Ketner, 1996, I shall adopt a version of the latter method of depicting a series of inferences. In so doing, I shall repeat the token for the SA at each moment of the inference, that is, at each step at which a transformation rule is applied. So, for example, let us see how the *modus ponens* example would look:

1.



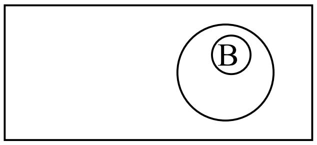
Premises

2.



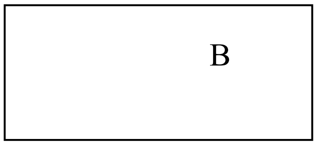
From 1 by DE

3.



From 2 by E

4.



From 3 by DC

X. Soundness Proofs

A. Basis cases.

Let ‘D |→ D*’ denote a single application of a transformation rule to D to derive D*.

1. For Insertion:

To Prove: If D |→ D* by Insertion, then D |= D*

Proof:

By Insertion, we may place any well-formed graph on an area of magnitude 2*n*-1 in diagram D. Let this graph be named ‘<I>’, and the diagram resulting from the application of Insertion to D, ‘D*’.

We must show that T_D covers T_{D*}. By the area reduction theorem for areas of odd magnitude, we find in T_{D*} a level at which a DD-type diagram occurs containing just one cut and inside the cut a set of diagrams A with <I> ∈ A. By successive application of Disjunctive Decomposition, at some further level we produce a tine of the singleton diagram cut-<I>. Note that all branches not passing through the node at which cut-<I>appears will retain exactly the same atomic information as do their obvious counterparts in T_D. Those branches passing through cut-<I>, on the other hand, will acquire additional atomic information from the further decomposition (if any) of the diagram cut-<I>.

Now consider a completed open branch of T_D. Because of the uniqueness of tree decomposition, and because cut-<I> branches off at a fork in T_{D*}, we can find a counterpart open branch in T_{D*} by matching the open branch in T_D with the open branch in T_{D*} which terminates with the same atomic diagram (note that this branch is not necessarily specified uniquely by the atomic letter and whether or not a cut appears, but is specified uniquely if we order instances of application of a particular decomposition rule in those cases where multiple instances of the same diagram appear at some level. In any case, this does not bear on finding a counterpart branch for covering). In all cases the matching branch contains the same atomic information as its counterpart in T_D, and so T_D covers T_{D*}.

2. For Erasure:

To Prove: If D |→ D* by Erasure, then D |= D*

Proof:

By Erasure, we may delete any well-formed graph that lies on an area of magnitude 2*n*-2 in diagram D. Let this graph be named '<E>' and the diagram resulting from the application of Erasure to D, 'D*'.

We must show that T_D covers T_{D*}. By the area reduction theorem for areas of even magnitude, we find a handle in T_D that includes <E>, whereas this instance of <E> is missing in T_{D*} (note: there may be other instances of <E> appearing in T_{D*}, but if so they will have counterparts in T_D as well).

Now consider a completed open branch of T_D. Either this branch passes through the handle which includes <E> or it does not. If it does not, then there is a counterpart branch in T_{D*} with the same atomic information. If it does pass through the handle containing <E>, then there will be (at least) one completed open branch of T_{D*} which contains all and only the atomic information exhibited on this branch up to and including the level at which the handle appears, with the exception of <E> if <E> is atomic. Whether <E> is atomic or not, this branch of T_{D*} will satisfy the covering requirement. We have now treated all cases and are done.

3. For Iteration:

To Prove: If D $|\rightarrow$ D* by Iteration, then D $|=$ D*

Proof:

By Iteration, we may take a token of any well-formed graph α that lies on an area of magnitude *m* within a nest of cuts in diagram D and scribe another token of that graph on any area of magnitude $\geq$ *m* within that same nest of cuts. Let this additional token be named '<C>' and the diagram resulting from the application of Iteration to D, 'D*'.

We must show that T_D covers T_{D*}. We will consider separately the cases in which <C> lies in an odd and an even area of magnitude.

Case 1. <C> lies in an area of magnitude 2*n*-1. In this case the proof follows the proof for Insertion given above.

Case 2. <C> lies in an area of magnitude 2*n*-2. Here we must consider subcases where the original token of α (which we will simply call 'α') lies on an odd or an even area of magnitude.

Case 2a. α lies in an area of magnitude *m* = 2*k*-2. In this case, by the area reduction theorem for even magnitudes, every branch which includes <C> will already include α. This means that no branches will contain additional atomic information as a consequence of <C>'s inclusion. More precisely: for each complete open branch of T_D, there is some complete open branch of T_{D*} with the same atomic information, so T_D covers T_{D*}.

Case 2b. α lies in an area of magnitude $m = 2k\text{-}1$. In this case, by the area reduction theorem for odd magnitudes, we find in T_{D*} a level at which a DD-type diagram occurs containing just one cut and inside the cut a set of diagrams A with $\alpha \in A$. By disjunctive decomposition applied successively, at some further level we produce a tine of the singleton diagram α surrounded by one cut. However, by the area reduction theorem for even magnitudes, by further decomposition applied to that $\kappa \in A$ such that <C> has been scribed within κ, we will eventually obtain a handle containing the graph <C>.

Since the decomposition tree T_D need contain no occurrences of <C>, it is clearly not the case that in all cases T_D covers T_{D*}; examples may be easily generated. Consequently, we need to introduce a new tree T obtained from T_D by applications of FDD. We do this as follows: Let κ* be the counterpart diagram to κ except without the inscription of <C>. At the bottom of every branch passing through κ* we apply FDD with the graphs <C> and cut-<C>. Following these applications, decompose <C> and cut-<C> in case <C> and/or cut-<C> are not atomic, and call this new tree T.

Claim: T covers T_{D*}. If we take a complete open branch in T that does not pass through (the token) κ, then we may find a copy of this branch in T_{D*}, so this case poses no problem. Now suppose we consider a complete open branch in T that does pass through (the token) κ. In this case the branch will either terminate in a decomposition of <C> or cut-<C>. In the former case we may find a branch in T_{D*} with the same atomic information, and we are done. In the latter case the branch will contain atomic information accruing from the process of decomposing cut-<C> plus any atomic information accrued prior to reaching κ. But there must be a path with just this information in T_{D*} as well, and we are done.

4. For Deiteration

To Prove: If D |→ D* by Deiteration, then D |= D*

Proof:

By Deiteration, for two tokens of graph α such that one token lies on an area of magnitude *m* within a nest of cuts in diagram D and the second token lies on an area of magnitude ≥ *m* within that same nest of cuts, we may erase the second token. Let the second token be named '<D>' and the diagram resulting from the application of Deiteration to D, 'D*'.

As in the proof for Iteration, we must consider distinct cases.

Case 1. <D> lies in an area of magnitude 2*n*-2. In this case the proof follows the proof for Erasure given above.

Case 2. <D> lies in area of magnitude 2*n*-1. Here we must consider subcases where the original token of α (which we will simply call 'α') lies on an odd or an even area of magnitude.

Case 2a. α lies on an area of magnitude *m* = 2*k*-2. In this case, by the area reduction theorem for even magnitudes, every branch of T_D which includes cut-<D> will already include α and hence will close. All other branches will have counterparts in T_{D*} with the same atomic information, and so we will have that T_D covers T_{D*}.

Case 2b. α lies on an area of magnitude *m* = 2*k*-1. In this case, by the area reduction theorem for odd magnitudes, we find in T_D a level at which a DD-type diagram occurs containing just one cut and inside the cut a set of diagrams A with $\alpha \in$ A. By Disjunctive Decomposition applied successively we produce a tine of the singleton diagram α surrounded by one cut. As before, let $\kappa \in$ A be that diagram from which the token <D> is removed in D*, and let κ* be the counterpart diagram to κ except without the inscription of <D>. Since <D> also lies in an area of odd magnitude, cut-<D> will be included along the decomposition of κ in D and will be absent in the decomposition of the counterpart κ* in D*. This does not place any additional constraint, then, on complete open branches of T_D covering complete open branches of T_{D*}, and since branches are otherwise identical, covering will always be possible, and we are done.

5. For Double Cut

To Prove: If D |→ D* by Double Cut, then D |= D*

Proof:

By Double Cut we may either insert a double cut around any graph on any area or remove a double cut that is around any graph on any area in diagram D. Call the diagram resulting from the application of Double Cut to D, 'D*'. Any T_D covers itself. The only difference between D and D* is that D* either includes an additional double cut around some graph or has one less double cut around some graph.

Case 1. A double cut is added. T_D covers T_{D*}, since the only difference relevant for covering between the trees is that T_D has a longer handle with no change to the set of atomic graphs included within the set of branches that include that handle. But by the semantic entailment definition, the covering relationship between T_D and T_{D*} is not affected.

Case 2. A double cut is removed. T_D covers T_{D*}, since the only difference relevant for covering between the trees is that T_{D*} has a longer handle with no change to the set of atomic graphs included within the set of branches that include that handle. But by the semantic entailment definition, the covering relationship between T_D and T_{D*} is not affected.

Since, in either case, T_D covers T_{D*}, D |= D*.

B. General case.

1. Definitions.

1. We say, in general, that A |-- B if there is a sequence $B_1, \ldots, B_n \ni A = B_1, \ldots, B_n = B$ and $\forall i\ i = 1, \ldots, n\text{-}1$, $B_i \mapsto B_{i+1}$. We call $B_1, \ldots, B_n$ an n-deductive chain.

2. Extended Coupled Trees: A set of three or more trees T_D, T_{D^*}, $T_{D^{**}}$, . . . , such that T_D covers T_{D^*}, T_{D^*} covers $T_{D^{**}}$, ...

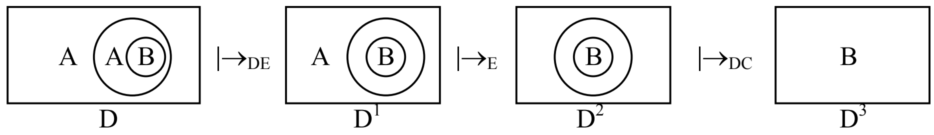
2. **Principle of Extended Coupled Tree Transitivity (PECTT)**: If T_D covers T_{D^*}, and T_{D^*} covers $T_{D^{**}}$, then T_D covers $T_{D^{**}}$.

Proof.

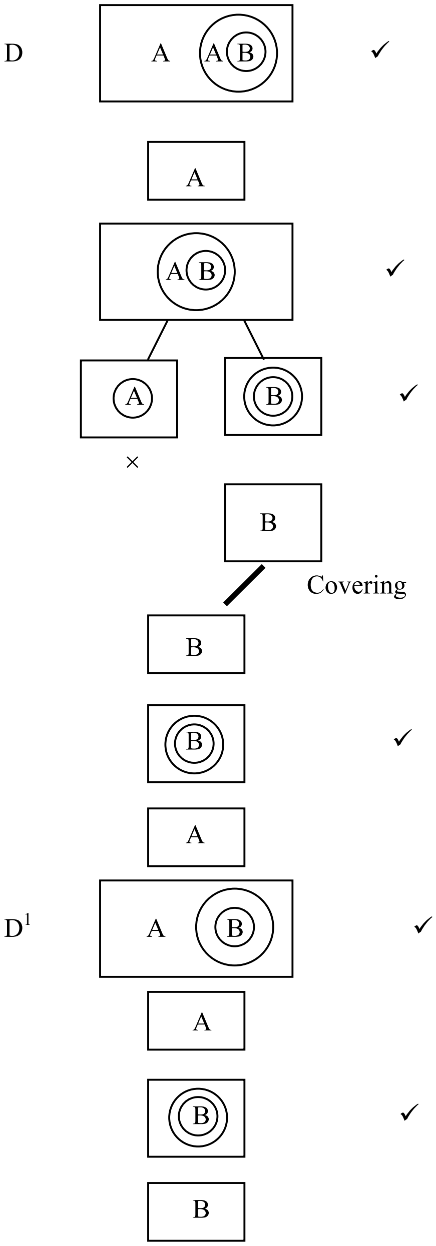
Assume T_D covers T_{D^*}, and T_{D^*} covers $T_{D^{**}}$. By the semantic entailment definition, $\forall b_D[b_D$ is an open branch of T_D , $\exists b_{D^*}(b_{D^*}$ is an open branch of $T_{D^*}) \ni (\alpha \in$ the set of atomic graphs of $b_{D^*} \rightarrow \alpha \in$ the set of atomic graphs of $b_D)]$ and $\forall b_{D^*}[b_{D^*}$ is an open branch of T_{D^*}, $\exists b_{D^{**}}(b_{D^{**}}$ is an open branch of $T_{D^{**}}) \ni (\alpha \in$ the set of atomic graphs of $b_{D^{**}} \rightarrow \alpha \in$ the set of atomic graphs of $b_{D^*})]$. But then $\forall b_D[b_D$ is an open branch of T_D , $\exists b_{D^{**}}(b_{D^{**}}$ is an open branch of $T_{D^{**}}) \ni (\alpha \in$ the set of atomic graphs of $b_{D^{**}} \rightarrow \alpha \in$ the set of atomic graphs of $b_D)]$. So, T_D covers $T_{D^{**}}$.

Corollary: By repeated applications of PECTT, we may say in general that for $j < l < n$, If $T_D^{\,j}$ covers $T_D^{\,l}$, and $T_D^{\,l}$ covers $T_D^{\,n}$, then $T_D^{\,j}$ covers $T_D^{\,n}$.

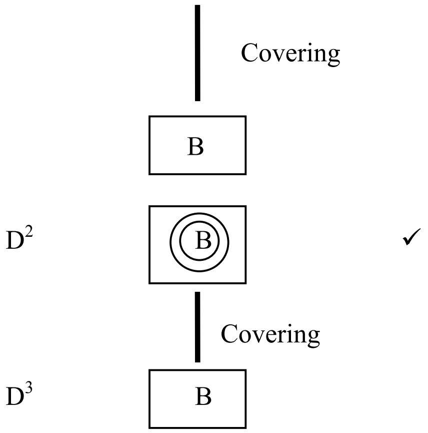
Illustration: Consider the following series of diagrammatic transformations:



and the extended coupled tree:



Observe the dual role of D^1: its tree exfoliating upward is covered by the tree of D that exfoliates downward; the tree of D^1 that exfoliates downward covers the tree of D^2 that exfoliates upward. There is a similar relationship for D^2 relative to D^1 and D^3.



3. Proof of general soundness:

For k = 1, already proved.

Assume k: i.e., D = D^1, . . . , D^k is a k-deduction chain. Assume D = D^1, . . . , D^{k+1} is a k+1-deduction chain. Since D = D^1, . . . , D^k is a k-deduction chain, so D |= D^k. By the result for k = 1, D^k |= D^{k+1} and so by transitivity, D |= D^{k+1}.

XI. Appendix 1: The “Alpha” Entitative Graphs

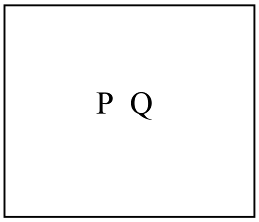
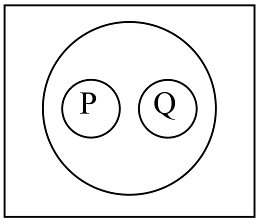
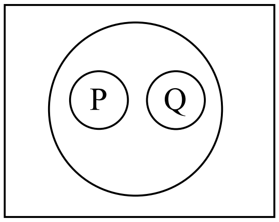
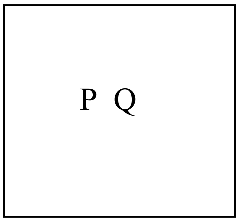
As I noted in the introduction, Peirce conceived of the Existential Graphs (which I will abbreviate as “EX”) in or about January 1897 while at work on his earlier Entitative Graphs (abbreviated “EN”). According to Ketner (1987), Peirce developed the principles of the existential system while composing his unpublished essay “On Logical Graphs” (Ms 482). Peirce described the Existential Graphs as “merely entitative graphs turned inside out.” (Ms 280, pp. 21-22, as quoted in Roberts, 1973, p. 27)

Now I should note here that Peirce never developed the various subsystems of EN as he did for EX. That is, where there are “alpha,” “beta,” and “gamma” EX, such subsystems do not exist in EN. Peirce’s writings of EN cover only what we might term “beta” graphs, that is, graphs pertaining to predicate logic. Thus, in what follows, I am speculating what *would* have constituted an alpha EN, should Peirce have worked out such a system.

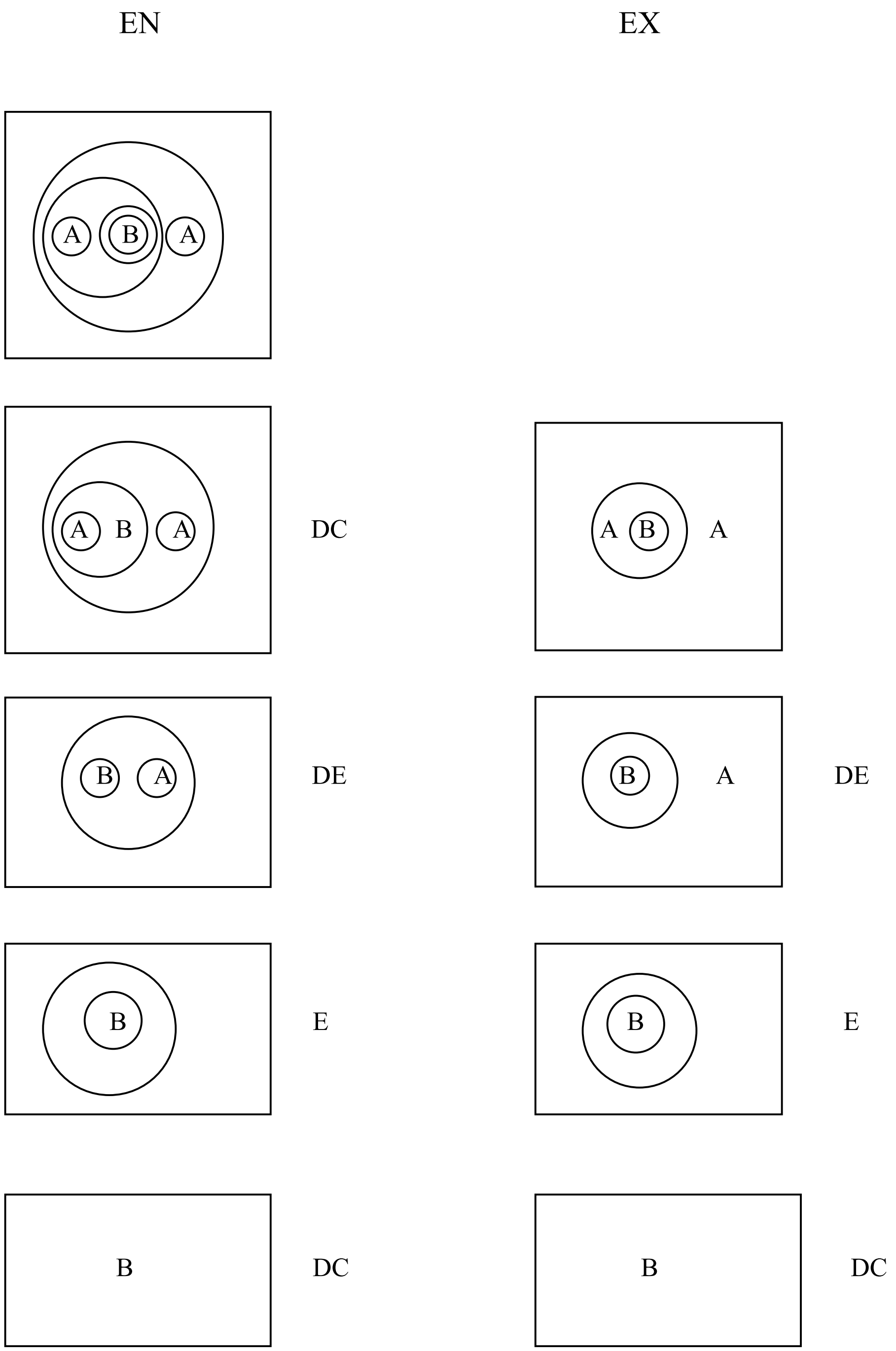
In his “On Logical Graphs” (Ms 482, §10 (p. 57), paragraph 1) he suggests a translation algorithm between the two systems. To take a diagram of EX and translate into EN (or vice-versa):

1. Write a cut around every sentence letter.
2. Around the *entire* set of graphs resting on the SA, write a cut around all of them.

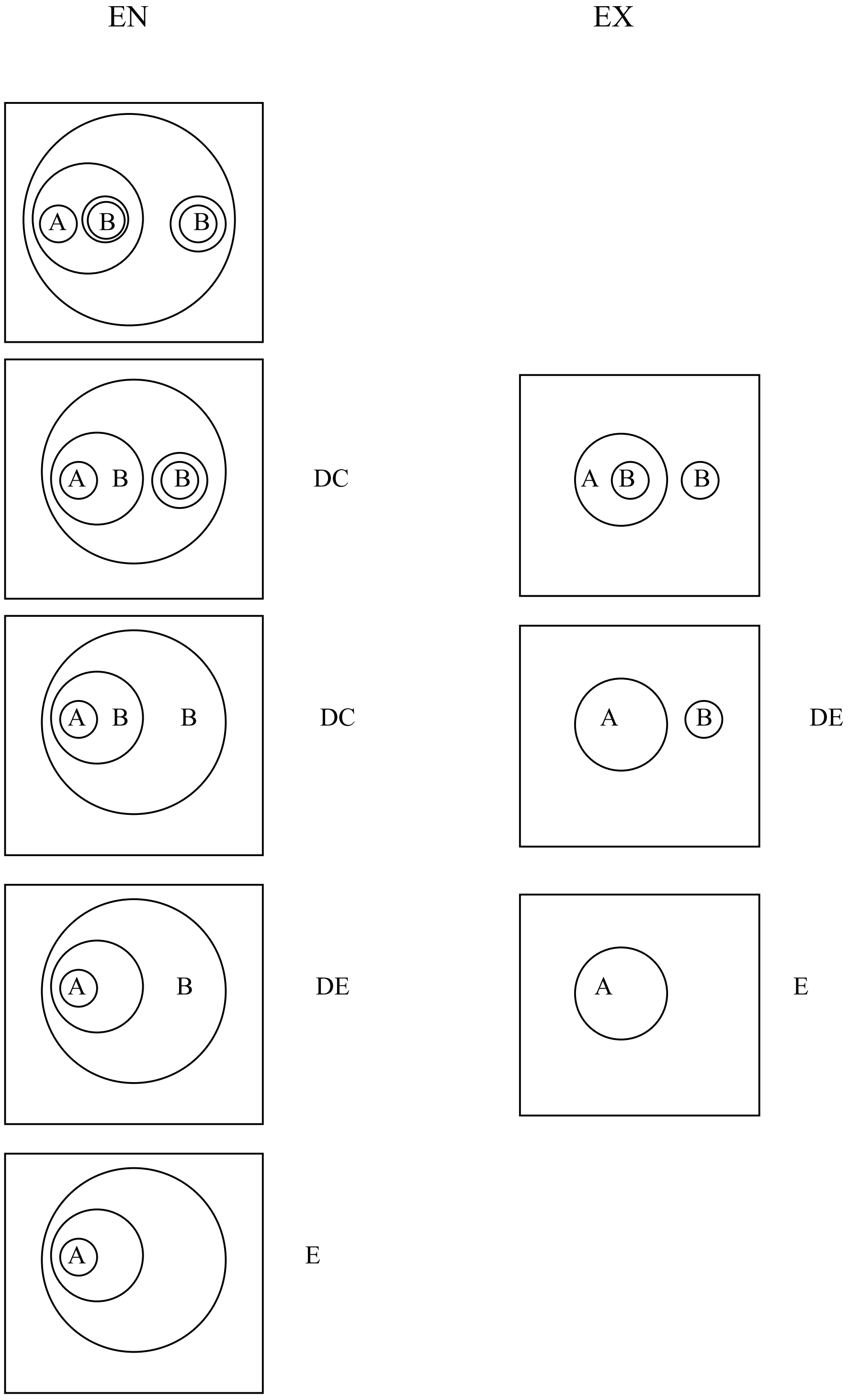
We may see the results of this algorithm in the following table (cf. Roberts, 1973, Appendix 2, p. 136):

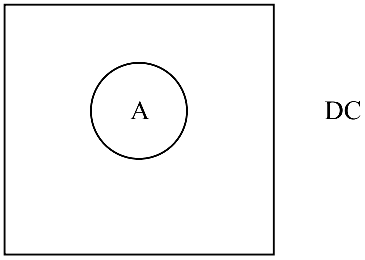
Read as:	P or Q	P and Q
EN:		
EX:		

This semantic difference should make a difference in the transformation rules. In his "On Logical Graphs," Peirce alludes to the transformation rules of the "beta" EN, but neither he nor any other writers on his graphical logic have worked out what the alpha EN rules would be. I suggest that the rules for EN are identical to EX, except that the two rules that refer to odd versus even areas. Specifically, the Insertion rule for EN should allow us to write any graph we please onto an area of *even* magnitude; the Erasure rule for EN should allow us to delete any graph we please from an area of *odd* magnitude. My suggestion can be illustrated by the following inferences, where on the left the inference is made within the EN system, and on the right the inference is made within EX. The first example is *modus ponens*:

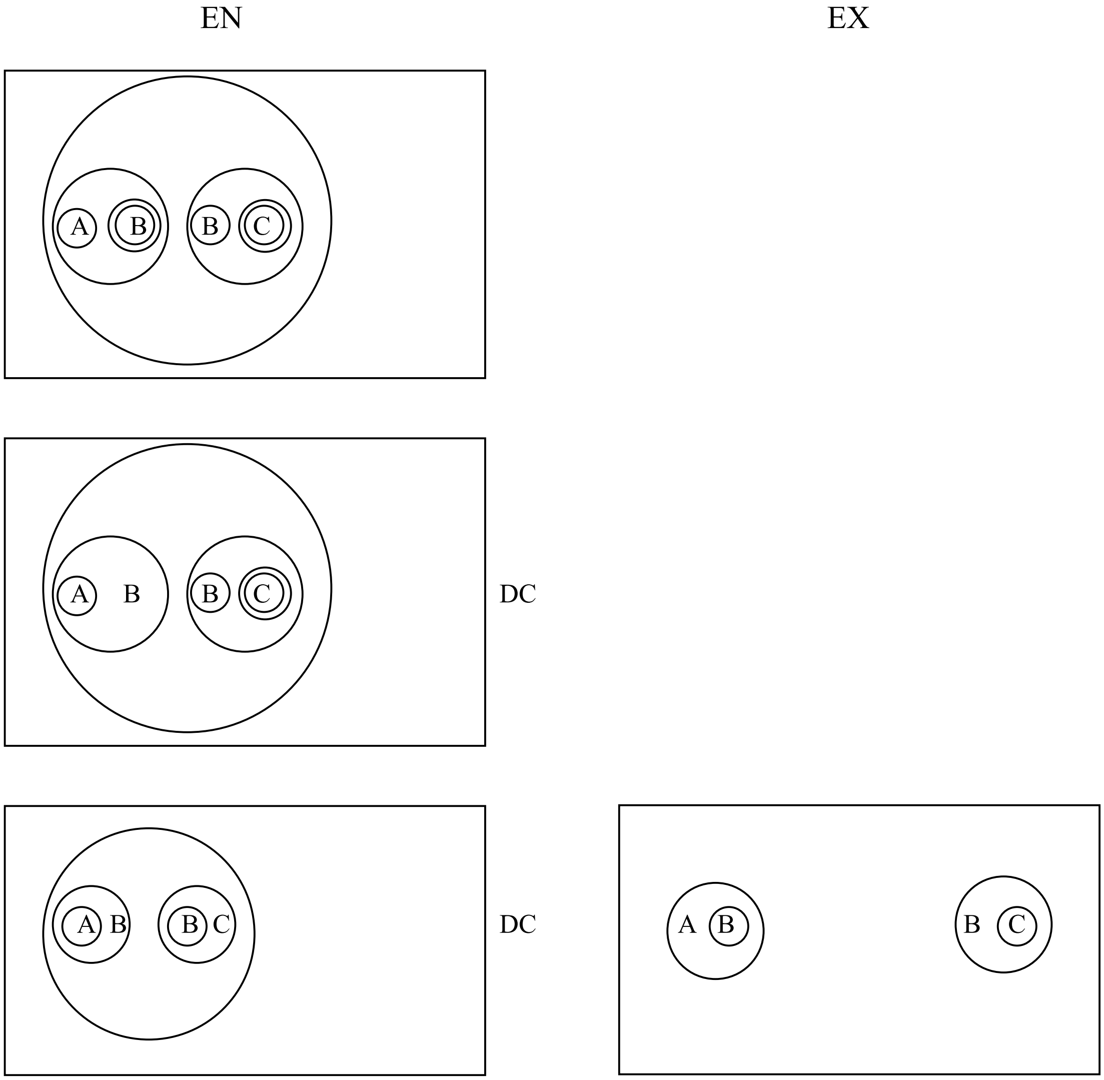

EN
EX
A
B
A
A
B
A
DC
A
B
A
B
A
DE
B
A
DE
B
E
B
E
B
DC
B
DC

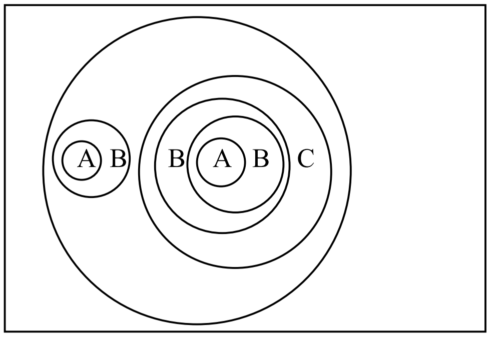
The second example is *modus tollens*:



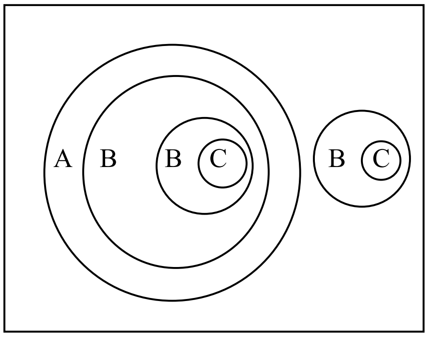


The third example will illustrate hypothetical syllogism:

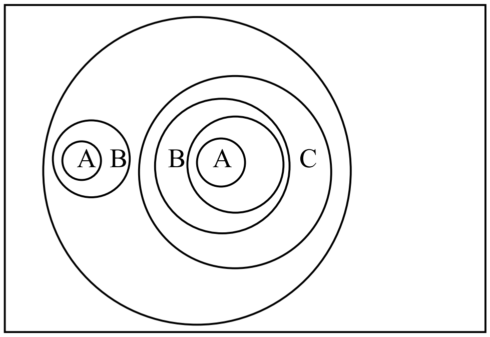




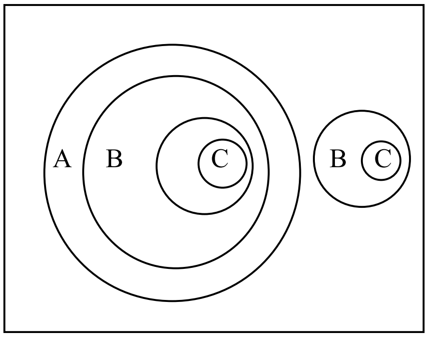
IT



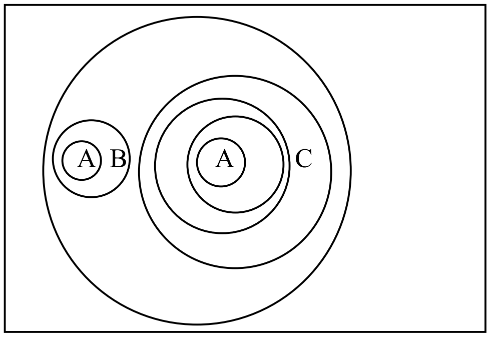
IT



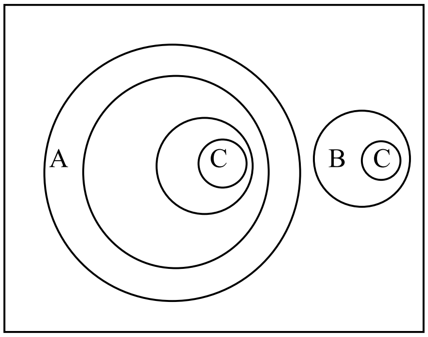
DE



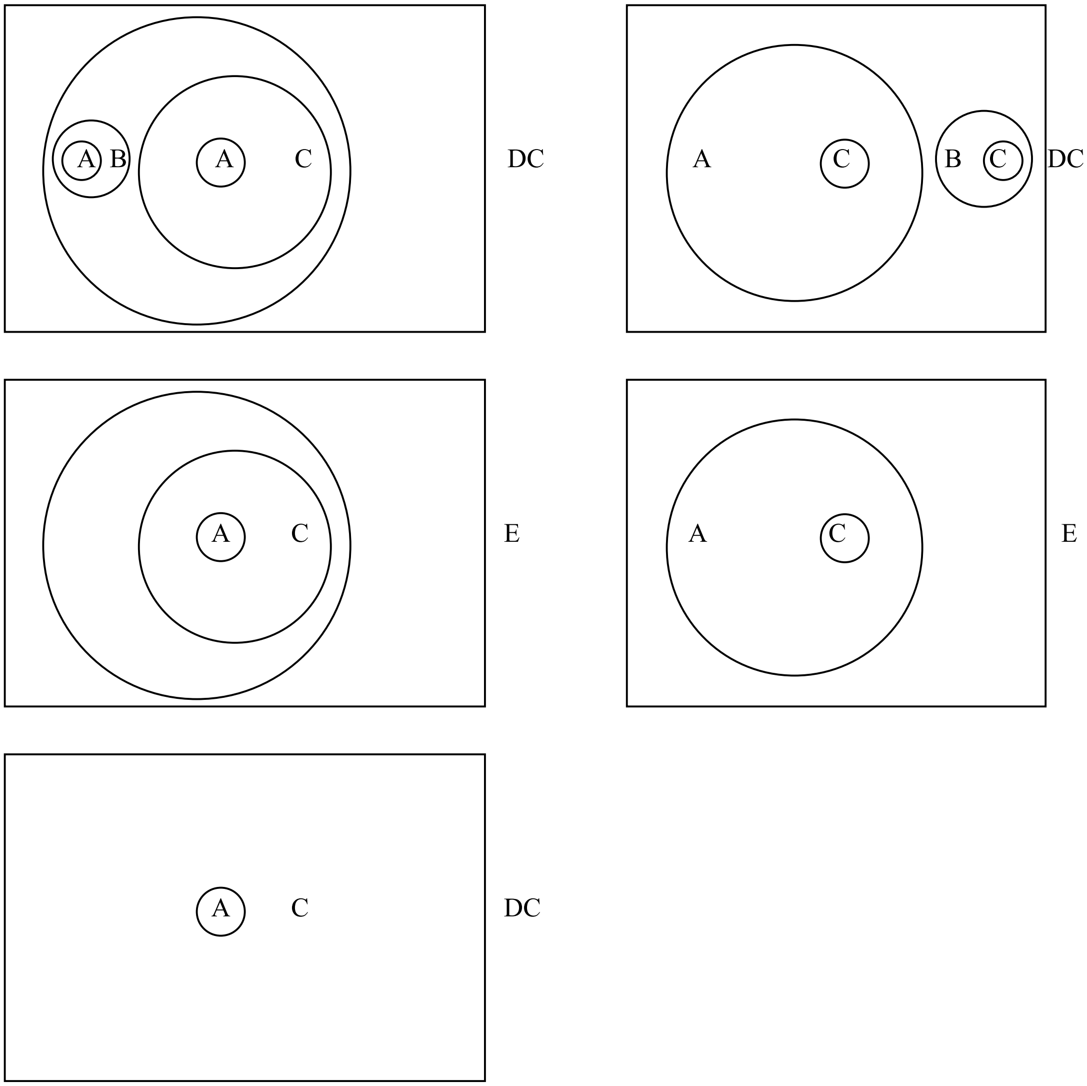
DE



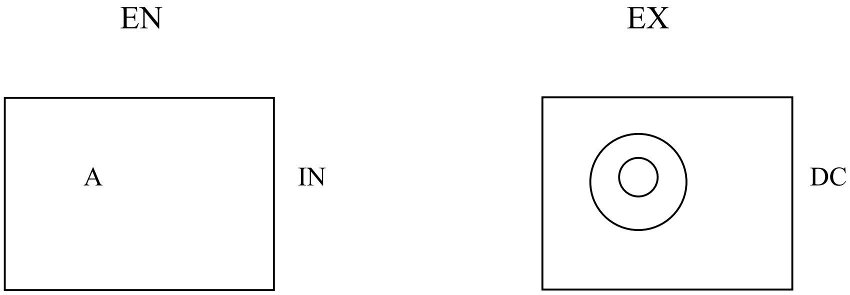
E

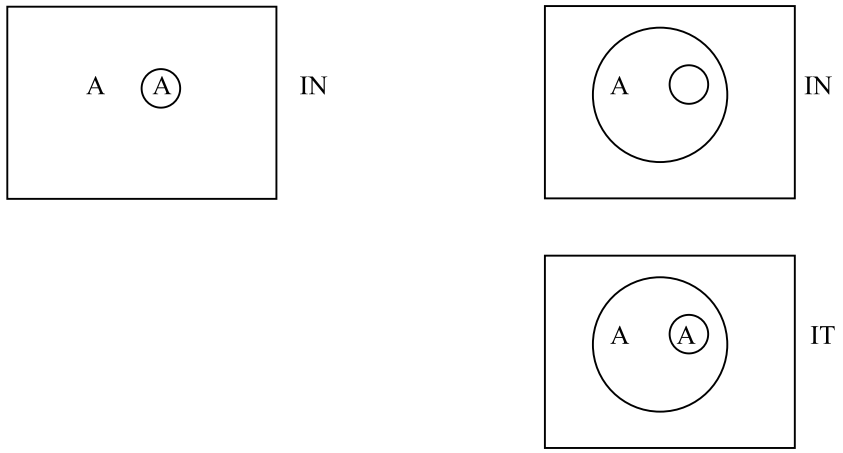


E



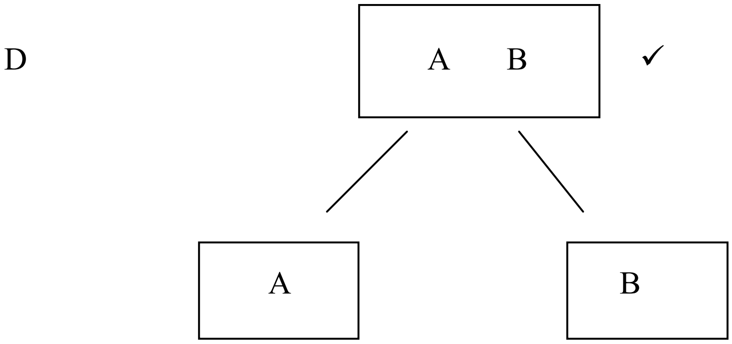
The final example will illustrate how to prove a theorem in both systems. In this case, the theorem is A ∨ ~A:





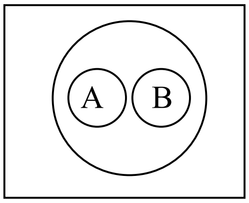
Finally I suggest that a soundness proof of alpha EN could follow the method I employed in the previous pages, but with the following major change to the decomposition principles:

For DD in EN, a tree should look like this:

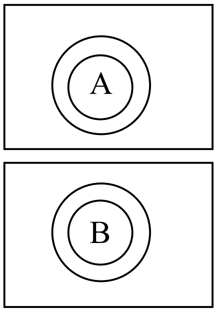


For CD in EN, a tree should look like this:

D



✓



XII: Appendix 2: Translating Algebraic Statements into Alpha Graphs

In thinking about how to formulate graphs for expressions in algebraic logic, I have found the following procedure to be very helpful.

First, convert the algebraic expression into the equivalent expression that uses only "~" and "&". DeMorgan's rule is crucial to this process. Here are some examples of this step:

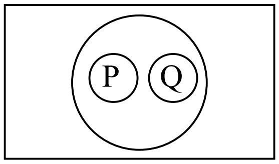
$(P \vee Q) \equiv (\sim\sim P \vee \sim\sim Q) \equiv \sim(\sim P \;\&\; \sim Q)$

$(P \supset Q) \equiv (\sim P \vee Q) \equiv (\sim P \vee \sim\sim Q) \equiv \sim(P \;\&\; \sim Q)$

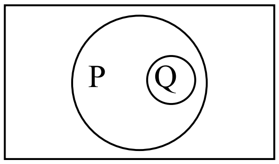
$(P \equiv Q) \equiv [(P \supset Q) \;\&\; (Q \supset P)] \equiv [(\sim P \vee Q) \;\&\; (\sim Q \vee P)] \equiv$
$\quad \equiv [(\sim P \vee \sim\sim Q) \;\&\; (\sim Q \vee \sim\sim P)] \equiv [\sim(P \;\&\; \sim Q) \;\&\; \sim(Q \;\&\; \sim P)]$

Second, from the outside in, convert "~" to a cut such that the scope of the "~" is the scope of the cut, and convert "&" into a juxtaposition of graphs within the enclosure of one cut. Here are some examples of this step:

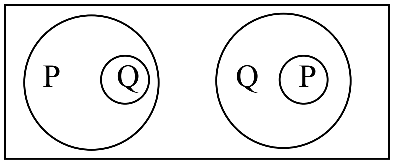
$(P \vee Q) \equiv \sim(\sim P \;\&\; \sim Q) \;\Rightarrow$



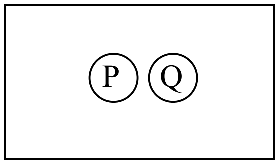
$(P \supset Q) \equiv \sim(P \;\&\; \sim Q) \;\Rightarrow$



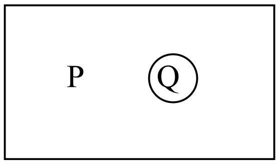
(P ≡ Q) ≡ [~(P & ~Q) & ~(Q & ~P)] ⇒



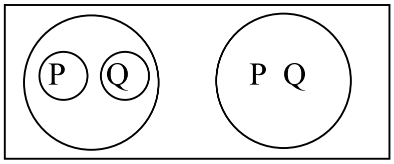
~(P ∨ Q) ≡ (~P & ~Q) ⇒



~(P ⊃ Q) ≡ P & ~Q ⇒



[(P ∨ Q) & ~(P & Q)] ≡ [~(~P & ~Q) & ~(P & Q)] ⇒



Bibliography

Judith Patricia Gail Butterworth, *The Modal System T in Peirce's Existential Graphs* [thesis] (Waterloo, Ontario: University of Waterloo, 1984).

J. Michael Dunn, "Intuitive Semantics for First-Degree Entailments and 'Coupled Trees'," *Philosophical Studies* v. 29 (1976), pp. 149-168.

Eric Hammer, "Peircean Graphs for Propositional Logic," in Gerald Allwein and Jon Barwise, eds. *Logical Reasoning with Diagrams* (New York, New York: Oxford University Press, 1996), pp. 129-147.

Richard C. Jeffrey, *Formal Logic: Its Scope and Limits* (New York, New York: McGraw-Hill Book Company, 1967).

Beverley Kent, "The Interconnectedness of Peirce's Diagrammatic Thought," in Nathan Houser, Don D. Roberts, and James Van Evra eds. *Studies in the Logic of Charles Sanders Peirce* (Bloomington, Indiana: Indiana University Press, 1997), pp. 445-459.

Kenneth Laine Ketner, *Elements of Logic: An Introduction to Peirce's Existential Graphs* (Lubbock, Texas: Arisbe Associates, 1996).

Kenneth Laine Ketner, "Identifying Peirce's 'Most Lucid and Interesting Paper'," *Transactions of the Charles S. Peirce Society*, v. 23 no. 4 (Fall 1987), pp. 539-555.

Charles Sanders Peirce, The Charles S. Peirce Papers. Manuscript collection in the Houghton Library, Harvard University, Cambridge, Massachusetts. n.d.

Charles Sanders Peirce, *Collected Papers of Charles Sanders Peirce*, 8 vols. Edited by Charles Hartshorne, Paul Weiss, and Arthur Burks (Cambridge, Massachusetts: Harvard University Press, 1931-1958).

Charles Sanders Peirce, "The Logic of Relatives," *The Monist* v. 7 no. 2 (January 1897), pp. 161-217.

Don D. Roberts, *The Existential Graphs of Charles S. Peirce* (The Hague: Mouton & Co. N.V., 1973).

Don D. Roberts, “A Decision Method for Existential Graphs,” in Nathan Houser, Don D. Roberts, and James Van Evra eds. *Studies in the Logic of Charles Sanders Peirce* (Bloomington, Indiana: Indiana University Press, 1997), pp. 387-401.

Sun-Joo Shin, *The Iconic Logic of Peirce’s Graphs* (Cambridge, Massachusetts: MIT Press, 2002).

John Stewart, *Theorem Proving Using Existential Graphs* [thesis] (Santa Cruz, California: University of California Santa Cruz, 1996).

J. Jay Zeman, *The Graphical Logic of C. S. Peirce* [diss.] (Chicago, Illinois: The University of Chicago, 1964).

J. Jay Zeman, “Peirce’s Graphs--the Continuity Interpretation,” *Transactions of the Charles S. Peirce Society*, v. 4 no. 3 (1968), pp. 144-154.