Hinged Sets And the Answer to the Continuum Hypothesis

by R. Webster Kehr

1st Version: January 20, 1999

**Dedicated to: Marit Olaug Liset Kehr,
my wife of 27 years and mother of our 7 children
and grandmother to our 6 grandchildren**

Abstract:

This paper introduces one of the most powerful proofs yet devised that N and R cannot be placed into bijection: the "Long N Theorem." However, using three new definitions of "countable," which are just as simple and just as logical as the definition of "countable" that Cantor used in his famous Diagonalization Theorem, this paper will also introduce two equally powerful proofs that R is countable. This paper deals with both sides of the issue of the cardinal number of R.

The concept of “cardinal number” is a concept of “size.” If two sets have the same number of elements, then they have the same cardinal number. If two sets can be placed into bijection then clearly they are the same size and have the same cardinal number. However, the question arises that if two sets with vastly different properties cannot be placed into bijection, is this a proof that these two sets have different cardinal numbers?

While it is true that N and R cannot be placed into bijection, this paper demonstrates that the reasons N and R cannot be placed into bijection have nothing to do with their cardinal numbers. The reasons have to do with the properties of irrational numbers and paradoxes related to the properties of R, such as the Ladder Paradox and the Set R1 Paradox. This paper proves that even if N and R had the same cardinal number, there would not be a bijection between them. The concepts leading up to this proof are very difficult to understand at first and are the main reason this paper is so long.

The new definitions introduced in this paper were designed to avoid paradoxes related to the properties of the elements of R. These three definitions determine the cardinality of a set based on an algorithm that can be used to create its elements.

In the process of resolving the dilemmas caused by Cantor's proofs, and the main proofs in this paper, the Continuum Hypothesis will be solved and all ancillary issues related to the Continuum Hypothesis will be answered.

INDEX:

Chapter 1: The Standard Definition of Countable:

Definitions: (Note: because of my emphasis on infinite sets my definitions are sometimes slightly different than standard definitions):

Let "R" be the set of all real numbers

Let "R(0,1)" or "(0,1)" be the real numbers between 0 and 1.

Let "N" be the set of natural or counting numbers, beginning with 0 or 1.

"N/o" - my ASCII symbol for "aleph nought," the cardinal number of N.

"Countable" - a set is countable iff it is the same "size" as N, meaning it has the same "number of elements" as N. The cardinal number of N is N/o, which is defined to be the smallest infinite cardinality.

(**Note:** in this paper the term "countable" applies **only** to infinite sets; thus a set is either finite, or if it is infinite, it is either countable or uncountable (i.e. uncountable means its cardinal number is larger than N/o). This means that in this paper the term "countable" and the term "infinite and countable" mean the same thing.)

"Mapping" - In this paper every mapping is a one-to-one mapping, even inverse mappings. Other types of mappings are not even considered.

"Onto" - An "onto" mapping from A onto B is a surjection, meaning a mapping from the elements of A onto all of the elements of B (i.e. an element of B cannot be found that is not mapped to by an element of A).

"Into" - An "into" mapping from A into B is an injection, meaning a mapping from the elements of A into some, but not necessarily all, of the elements of B. (Note: in this paper an injection can map to some or all of the elements of a set - this of course is slightly different than some authors but it is necessary to more easily prove a bijection when working with infinite sets). A "pure into" mapping is a mapping from A into some, but not all, of the elements of B.

"Bijection" - A bijection between two sets, A and B, means there is a one-to-one correspondence between the elements of A and the elements of B. This can be proven by demonstrating that A maps "into" B and B maps "into" A or in other ways. (Schroeder-Bernstein Theorem)

A "finite expansion," "finite decimal" or "terminating decimal" has a finite number of significant digits (e.g. 1/2, .485, etc.).

An "infinite expansion," "infinite decimal" or "nonterminating decimal" is an element of R that has an infinite number of significant digits (e.g. pi/10, e/pi, 5/9, etc.).

"Set Rt" - the set of terminating decimals in R(0,1).

"Set Rnt" - the set of nonterminating decimals in R(0,1).

"Digit" or "nth digit" means: "the character, taken from the set or pool: {0, 1, 2, ..., 9}, in the nth position of the number (counting positions from the leftmost character consecutively to the right and ignoring decimal points); meaning the nth consecutive significant digit from the left." For example, the 6th "digit" of pi/10 is '9' (i.e. .31415'9'...).

In essence the terms "digit" and "character" both refer to an element of the set {0, 1, 2, ..., 9}, however, when the term "digit" is used it implies the use of an index as part of the term. Thus, the set of "digits" in pi/10 is countable (i.e. infinite and countable), but the set of "characters" in pi/10 has a cardinality of 10.

"Digit Position Index" (DPI): The "nth digit" of an element of R(0,1) is assigned an "index" number, which is an element of N, called the Digit Position Index (DPI). For example, the 6th digit of pi/10 is a '9' (i.e. .31415'9'...), thus this '9' is in the 6th DPI of pi/10. "Digit," "nth digit," "digit position" and DPI all refer to the same position. A DPI by definition is an element of N.

"ISD" - in this paper every element of R(0,1) will frequently be viewed as an "Infinite String of Digits" or ISD, including both finite and infinite expansions in R(0,1). An ISD simply means that every element will be considered to have an infinite number of digits, each of which is significant. This, of course, means that all zeros are significant.

For example, 1/2 will equal .50000... Using this terminology R(0,1) can be looked at as the set of all possible permutations, with redundancy, of ISDs. Thus, "R(0,1)" and "the set of all possible ISDs in R(0,1)" are two different ways to describe the same set. The two phrases: "each element of R(0,1)" and "each ISD of R(0,1)" will be used interchangeably. Note that .50000... is a perfectly valid permutation! Note also that every ISD has exactly the same set of DPIs: N.

A Brief Overview of the Diagonalization Theorem:

By far the most famous proof that R(0,1) (and thus R) is uncountable is Georg Cantor's Diagonalization Theorem (DT) of 1891. For a more thorough explanation of the background of his proof and his definitions, than I will present, the reader should study: ***Georg Cantor - His Mathematics and Philosophy of the Infinite***, by Joseph Warren Dauben (Princeton University Press - 1979).

There are several different ways to prove the DT. I will provide a brief overview of one type of proof to establish some of the terminology I will use.

In a proof by contradiction, the DT begins by assuming that R(0,1) is countable. It then states that if R(0,1) were countable it could be listed side-by-side with N (i.e. there would exist a bijection between N and a listing of R(0,1)). A representative listing of a countable number of elements of R(0,1) is shown (i.e. "Given any countable subset of R(0,1)") and an effort is made to determine whether all of the elements of R(0,1) can be in this countable listing.

Once the listing is shown, a technique known as diagonalization creates (or discovers) an element of R(0,1) which cannot be mapped to by a specific element of N. This element of R(0,1) I call the "new number" or NN. Since no specific element of N maps to the NN it is concluded that this listing cannot contain every element of R(0,1), meaning R(0,1) cannot be placed into a countable listing. This is a contradiction and it is concluded that the original assumption, that R(0,1) is countable, is false and that therefore R(0,1) is uncountable.

The Use of the "iff" Symbol In Definitions:

The DT uses an “iff” definition of “countable” in its proof. Before studying this definition, let us study what the use of the “iff” symbol implies.

Consider this definition of a "pregnant" box turtle (i.e. a box turtle soon to be ready to lay eggs):

Definition: "A box turtle is pregnant iff it is a female."

This "iff" definition can be split up into two "if" definitions:

1st "if" Definition: "if a box turtle is a female then it is pregnant."
2nd "if" Definition: "If a box turtle is pregnant then it is a female."

The first "if" definition is false because not every female box turtle is pregnant. Some are too young, and so on. Since one of the "if" definitions is false then the "iff" definition is also false. It is clear that the two "if" definitions are **logically independent** because the second one is true but the first one is false.

The first "if" statement can also be reworded like this: "the set of all female box turtles is a subset of the set of all pregnant box turtles." This is false. The second "if" statement can be reworded like this: "the set of all pregnant box turtles is a subset of the set of all female box turtles." This is true. Taken together, the "iff" statement is this: "the set of all female box turtles is equal to the set of all pregnant box turtles." This is false.

The use of the "iff" symbol in a definition implies the absolute equality of two independent sets, in this case the set of all female box turtles and the set of all pregnant box turtles. Since the set of all pregnant box turtles is a proper subset of the set of all female box turtles the two sets are not equal and the use of the "iff" symbol is false.

One last note, if the universal set is defined to be the set of all male box turtles union the set of all pregnant box turtles then the above definition is true. However, if the universal set is defined to be the set of all box turtles, then the definition above is false.

The Standard Definition of "Countable":

Now consider the Standard Definition of "countable" used in most textbooks:

Standard Definition (SD): "A set is countable iff it can be placed into bijection with N."

As above, this "iff" definition can be split up into two "if" definitions:

1st "if" Definition: "If a set can be placed into bijection with N, then it is countable."
2nd "if" Definition: "If a set is countable then it can be placed into bijection with N."

Obviously, the first "if" definition is true. In this paper I will call this definition the "Definition of Countable" or DOC.

The second "if" definition, while logical, is **logically independent** of the first. In other words, the obvious validity of the first "if" definition does not automatically imply the validity of the second "if" definition.

The contrapositive of the statement: "a implies b" is the statement: "not b implies not a." Any statement and its contrapositive are logically equivalent. If the second "if" statement above were the contrapositive of the first "if" statement above then the SD would be automatically true; however, by the nature of the "iff" symbol they are not contrapositives of each other.

The contrapositive of the first "if" statement is this: "if a set is not countable, then it cannot be placed into bijection with N." This statement is logically equivalent to the first "if" statement above, the DOC, and is equally obviously true.

The contrapositive of the second "if" statement is this: "if a set cannot be placed into bijection with N, then it is uncountable." In this paper I will call this statement the "Definition of Uncountable" or DOU.

With the DOU now defined the second "if" statement in the SD now becomes the "Contrapositive of the DOU," or CDOU. The SD thus consists of two logically independent "if" statements: the DOC and the CDOU. I use this seemingly strange notation because of the logic of the DT. The DT actually uses the contrapositive of the DOU in its proof.

Thus the system of transfinite mathematics contains two logically independent definitions. The DOC is used to establish a "sufficient" condition to determine that a set is "countable." The DOU establishes a "sufficient" condition to determine that a set is uncountable. However, vastly different techniques and logic must be used in each case!

Because the DOC and DOU are logically independent it must be shown that these two definitions are consistent with each other, meaning they will always yield the same results. There is no proof of the absolute consistency of the DOC and DOU with respect to all mapping techniques and all possible infinite sets, no matter what the differences in properties between the two sets.

The use of the "iff" definition implies that there does not exist a set that can be proven to be both countable and uncountable using two different techniques. This paper will produce such a set. Using one of Cantor's theorems it will be shown to be countable, but using a definition consistent with the use of diagonalization, this set will be shown to be uncountable.

Another Look at the SD:

To state the SD another way: the SD assumes that for the universal set of all infinite sets, every set that can be placed into bijection with N and every set that is the same "size" as N are the same set. This implies that the DOC will "work" on every possible set that has the same "number of elements" as N, and vice versa. This is the basic assumption of the SD.

However, because the DT uses diagonalization, a major set of new assumptions is thrown into the equation. The SD assumes that diagonalization will "work" on every infinite set that is not that same "size" as N, and that diagonalization will "not work" on every possible infinite set that does have the same "number of elements" as N.

While it is obvious that if two infinite sets are in bijection, they have the same cardinal number, it is not so obvious that if diagonalization "works" on a set that it has a different "number of elements" as does N.

Note that it is assumed that if diagonalization "works" on a set, by definition the DOC does "not work" on the set. As will be shown, the use of diagonalization adds an enormous complexity to the seeming simplicity of the SD. It is this complexity that makes this paper so long!

Summary of Chapter I:

Because the DT uses an “iff” definition, it is actually using two logically independent definitions: the DOC and the CDOU. While the DOC is obviously true, because the CDOU is logically independent of the DOC, the CDOU is not automatically true.

Furthermore, because of the use of diagonalization it is not automatically true that the DOC will work on every possible set that is the same "size" as N. If diagonalization does "work" on a set, then the DOC, by definition, does "not work" on the set.

The SD actually goes far beyond these assumptions but these are the key assumptions this paper will deal with in its early sections.

Chapter II: The DOU/Diagonalization Combination

The Diagonalization Theorem (DT) in More Detail:

The DT starts out by stating: "given any countable listing of R(0,1)." It then places N in bijection with this set. The justification for doing this is the CDOU, which states that if a set is countable it can be placed into bijection with N. For example:

```
N    Symbol     A Countable Subset of R(0,1)

1    pi/10      .314159265 ...
2    5/9        .555555555 ...
3    1/2        .500000000 ...
4    1/e        .367879441 ...
5    3/4        .750000000 ...
...
```

At this point the DT asks the question: "can every element of R(0,1) be in this countable subset?" To answer this question the DT creates (or discovers) an element of R(0,1), the NN, which is not mapped to by any element of N. It does this with a sequence and algorithm. The standard algorithm used in this paper is this: "if the nth digit of the nth element of the list is not a '5', then the nth digit of the NN will be a '5,' otherwise the nth digit of the NN will be a '6.'

Using this algorithm the first digit of the NN is a '5' (note: the 1st digit of the 1st element above, pi/10, is a '3'). The second digit of the NN is a '6' (note: the 2nd digit of the 2nd element above, 5/9, is a '5'). The third digit of the NN is a '5' (note: the 3rd digit of the 3rd element above, 1/2, is a '0'), and so on. The NN equals: .56555...

At this point the DT uses the DOU. Since no specific element of N taken from the listing maps to the NN, therefore N cannot map onto all of the elements of the listing, therefore R(0,1) cannot be placed into bijection with N. Therefore, by using the DOU it is concluded that R(0,1) is uncountable.

Note that the DT depends entirely on the CDOU and the DOU. The DT uses the CDOU to set up the original assumption that if R(0,1) were countable it could be placed into bijection with N. It uses the DOU when it is determined that no specific element of N maps to the NN.

More on Diagonalization:

The DT can be broken down into two parts:

1) the use of diagonalization proves that N and R(0,1) cannot be placed into bijection, and
2) the DOU states that if N and R(0,1) cannot be placed into bijection then R(0,1) must be uncountable.

If the DOU were not available to the DT, diagonalization, by itself, is not sufficient to prove that R(0,1) is uncountable.

To understand this let us look at an attempt to "list" **all** of the elements of R(0,1). If we consider all of the elements of R(0,1), diagonalization divides R(0,1) into two disjoint sets: 1) the set of all elements to which N can map, and 2) the compliment of the first set. The NN is in the "second set," the compliment. By creating the NN diagonalization proves that the compliment of the first set is not the null set.

In fact diagonalization obviously proves that the compliment is infinite. But "infinite" does not automatically imply uncountable. Countable sets are also infinite.

The key to determining the cardinality of R(0,1) is to determine the cardinality of the second set, the compliment. If the second set is countable, then by Cantor's CUT (i.e. the CUT is Cantor's Countable Union Theorem which proves that the union of a countable number of countable sets is itself countable) all of R(0,1) is countable. If the second set is uncountable, then obviously all of R(0,1) is uncountable.

The DOU basically states that if the second set is not the null set, then it must be uncountable. It is the **combination** of diagonalization (which only proves that the compliment is infinite) and the DOU (which states that if the compliment is not the null set it must be uncountable) that determines that R(0,1) is uncountable.

Given any countable listing of R(0,1) elements, a NN can be constructed. But the question is this: "by using different algorithms, how many unique NNs can be constructed?" The DOU says that an uncountable number can be created.

The DOU is a definition, not a mapping technique. The DOU, by itself, does not prove anything. The DOU must be paired with a mapping technique. The DT pairs the DOU with diagonalization. Thus the **combination** of the DOU and diagonalization must be valid.

By using the DOU/diagonalization combination the DT therefore assumes that the DOU/diagonalization combination cannot "work" (i.e. prove the lack of a bijection with N) on any countable set.

What Does the DT Prove?

The question becomes: "what does the DT prove?" The answer depends on which assumption you start with.

If one starts the DT with a belief that the DOU is true and that R(0,1) is assumed to be countable, then diagonalization creates a contradiction and it is concluded that the original assumption is false, meaning the original assumption that R(0,1) is countable is false.

On the other hand, if one starts with a belief that R(0,1) is countable and that it is assumed the DOU is valid (when used with diagonalization), then the contradiction arrived at with the use of diagonalization proves that the original assumption is false, namely the assumption that the DOU is valid when used with diagonalization.

The DT claims that the DOU is true, thus it does not claim to begin with two assumptions. But in fact because the DOU is logically independent of the DOC, and just as importantly because the DOU is used in conjunction with diagonalization, the DT begins with two assumptions, namely: 1) the DOU is true (in combination with diagonalization) and 2) R(0,1) is countable.

(Note: From this point on in the paper, whenever I talk about the DOU, I am usually talking about the DOU/diagonalization combination. The reader should note context. Near the end of this paper the DOU will be discussed by itself.)

With those two assumptions, then what does the DT prove? The DT only proves that both of these assumptions cannot be true! Knowing that the DT does not prove that the DOU is true, this fact in and of itself is sufficient to prove that the DT falls short of proving that R(0,1) is uncountable.

The problem, of course, is whether the DOU should be assumed to be true, particularly knowing that the DOU must be paired with diagonalization. Why did the DT assume the DOU/diagonalization combination is true?

Dauben's book does not deal with this question, but when talking about other issues he states that the mathematical community intuitively believed that R was "bigger" than N long before 1891. The stage was actually set for the mathematical community to abandon the countability of R even before 1874. This intuitive belief was first satisfied by Cantor in 1874 when he developed an earlier proof of the different cardinalities of N and R using a form of nested intervals.

Thus when the DT concluded that R was uncountable by using the DOU, the community essentially concluded that the DOU was true because it gave them the answer they already believed. In other words, one could say that the intuitive assumption that R was uncountable led to the agreed-upon assumption that the DOU/diagonalization combination was true.

The point to this discussion is that the DOC and DOU/diagonalization combination are clearly logically independent and it has never been proven that the set of all sets which are the same "size" as N and the set of all sets in bijection with N are the same set. This means there is no proof that the DOC and DOU/diagonalization combination are consistent. The DT depends heavily on an unproven definition that has never been shown to be logically or mathematically consistent with the obviously valid definition - the DOC.

The paper will produce a proof that divides all of R(0,1) into two disjoint sets. Both sets will be shown to be countable.

Summary of Chapter II:

The DOC and DOU are logically independent. While the DOC is universally accepted as being true, since the DOU is not its contrapositive and because the DOU is logically independent of the DOC, and because the DOU is used with diagonalization, the DOU remains unproven, particularly when combined with diagonalization.

Since the DT depends totally on the CDOU and DOU it is also clear that the DT does not prove that R(0,1) is uncountable (unless the DOU is assumed to be true). The DT actually proves that two things cannot **both** be true: 1) the countability of R(0,1) and 2) the DOU/diagonalization combination is valid.

Chapter III: Array Orientations, the ACN, and Width

Definitions:

The "width" or "width set" of a **number** is the set of DPIs of the number. For example, the "width" or "width set" of .654124 is {1, 2, 3, 4, 5, 6} or by using the shorter notation: [6].

The "width" or "width set" of a **set** is a set of consecutive counting numbers, beginning with the number 1, which is equal to the union of all of the width sets of the individual elements of the set. For example, the width of this set:
{.89304, .3, .00732893558, .9048372} is [11].

"Complete Permutation Construction" (CPC) - A set S has a CPC if and only if given any element of S, say s, which has a width [w], then every possible permutation of w digits (base 10) is a unique and valid element of S.

For example, if R(0,1) is considered to be a set of ISDs, then R(0,1) has a CPC.

Set Rt, on the other hand, does not have a CPC. For example: .375 has a width of [3], but the permutation with three digits of ".400" is not a unique and valid element of Set Rt because it is redundant with .4, which has a width of [1]. Thus not every permutation of three digits is a unique and valid element of Set Rt. Thus Set Rt does not have a CPC.

"FIS" - Finite Initial String or Finite Initial Segment. Given any ISD, and given any DPI of this ISD, say n, the FIS is the first n digits of the ISD and has a width of [n]. For example, the 5th FIS of pi/10 is .31415 and has a width of [5].

The Concept of Orientation:

The concept of "diagonalization" or "diagonal" requires a two-dimensional array. How can you have a "diagonal" without having both rows and columns?

The use of diagonalization by the DT causes R(0,1) to be viewed as a two-dimensional array of characters taken from the pool: {0, 1, 2, ..., 9}. To understand this let us look at a countable listing of R(0,1) elements viewed as an array of characters and call it Set GACSR (where GACSR stands for "Given Any Countable Subset of R(0,1)"), which we will call Set G for short:

```
Set G:

   Column     1 2 3 4 5 6 7 8 9 ...  (Column Numbers)
Row

pi/10       . 3 1 4 1 5 9 2 6 5 ...
5/9         . 5 5 5 5 5 5 5 5 5 ...
1/2         . 5 0 0 0 0 0 0 0 0 ...
1/e         . 3 6 7 8 7 9 4 4 1 ...
3/4         . 7 5 0 0 0 0 0 0 0 ...
...
```

An array of this type has two different names, each of which serves a different conceptual purpose. First, this type of array is called a "Single Set Array" (SSA) because it views a single set as a two-dimensional array of characters.

Note that a SSA has two different infinite orientations: an infinite column orientation, which is in bijection with N by definition (i.e. the set of "column numbers" is N), and an infinite row orientation, which for Set G is also in bijection with N by definition. The term "orientation" is chosen because a SSA represents the characters of a single set, thus the columns and rows are two different views or orientations of the same set.

The column numbers are obviously called the **"width orientation"** and the row numbers are called the **"row orientation"** or "length orientation."

The second name for this type of array is a "character array." This name exists because each cell of the array contains a single character of a single element.

Without character arrays there would be no such thing as diagonalization. The term "diagonalization" refers to the diagonal cells of a character array, beginning from the top left corner of the array. The diagonal cells, for all n in N, are the cells where the nth column and the nth row intersect, meaning the cells or characters that meet the criteria: the nth digit of the nth element. For example, the cell in the 100th column and 100th row is a diagonal cell. The cell in the 100th column and the 56th row is not a diagonal cell.

The Axiom of the Counting Numbers (ACN):

While everyone knows there is only one definition of N, it is helpful to convert this common knowledge into a simple axiom in order to use this fact in proofs.

While most of the items in this axiom are obvious, the need for items #3a, #3b and #3c will become obvious as the paper progresses. For now the reader should simply focus on items #1 and #2.

The Axiom of the Counting Numbers (ACN):

Let sets Na and Nb be two sets of natural or counting numbers, where both sets meet all of the following criteria:

1) Both sets include all consecutive elements of N beginning with their first element, and
2) The first or smallest element of both sets is 0 or 1, and
3a) Both sets have the same orientation, or
3b) The two sets are not Hinged Sets, or
3c) If the two sets do not have the same orientation and they are Hinged Sets, then they are one-to-one synchronized Hinged Sets,

then sets Na and Nb both:

1) Have the same elements, meaning they are equal (i.e. they are the same set), and
2) There exists a bijection between Na and Nb

End of Axiom

While the reader might not realize it at this point in the paper, the ACN will be correctly used in all proofs.

Set Nv:

The elements of N are not designed for a permutation-based discussion. Consider this subset of N:

Set Nv = {x | x is an element of N and the first digit of x is a 5, except 5 itself}
Set Nv = {50, 51, 52, ..., 59, 500, 501, 502, ..., 599, 5000, ...}

Let us convert the initial 5 of each element of Set Nv into a 'v', the Roman Numeral for 5.

Set Nv = {v0, v1, v2, ..., v9, v00, v01, v02, ..., v99, v000, ...}

One purpose for this aesthetic conversion is that this paper will not treat the initial 'v' of each element of Set Nv as a significant digit (much like a decimal point is not a significant digit). For example, v8936504 has 7 significant digits and a width of [7]. Another reason for the conversion to 'v' is to distinguish the elements of Set Nv from the elements of other sets.

There are two main reasons for defining Set Nv:

1) Every zero of every element is significant, so that v007800 has 6 significant digits. Note that .007800 in Set Rt has only 4 significant digits (i.e. .0078) and note that 007800 in N also has only 4 significant digits (i.e. 7,800), and
2) Because every zero is significant Set Nv has a CPC.

Because of Set Nv's permutation nature, and the fact that it is a proper subset of N, there will be instances in this paper where Set Nv will be substituted for N.

The Nv Theorem:

Nv Theorem: For every element of N, n, there exist 10^n elements of Set Nv with a width of [n].

Proof:

Suppose there exists an element of N, n, for which the subset of Set Nv of width [n] did not have 10^n elements. The maximum number of possible permutations of n positions taken from a pool of c characters, with redundancy, is c^n. Because Set Nv is base 10, the subset of Set Nv of width [n] cannot have more than 10^n elements of width [n]. Thus, if false, for some n there must be less than 10^n elements of width [n].

If for some n the cardinality of Set Nv was less than 10^n, then v999...999 would not be an element of Set Nv, where the number of 9s represented by the three dots is n minus 6. Thus 5999...999 would not be an element of N, where the 'v' is converted back to a '5'. This means the cardinality of N must be less than 5999...999. But since n is finite, 5999...999 is a finite expansion, meaning N would be finite. This is a contradiction.
QED

Corollary: The width of Set Nv is N

Proof: If, for every element of N, say n, there exists an element of Set Nv with a width of [n], then by definition the width of Set Nv is N. This is because the union of all of the width sets of the elements of Set Nv would be N (i.e. the width set would include every element of N), which is the very definition of the "width" of a set. Suppose the width of Set Nv was not N. Then for some n, an element of N, no element of Set Nv would have a width of [n]. By the Nv Theorem this is a contradiction. **QED**

Saying that the width of R(0,1) is N (the set of column numbers of its SSA is N) and the width of Set Nv is N might be surprising to some because R(0,1) consists of infinite expansions and Set Nv does not contain a single infinite expansion. The reasons for their equal width will become clear as the paper progresses. In fact much of the rest of this paper deals with the reasons for, and ramifications of, their equal width.

Set N5:

Definition:
Set N5 = { x **|** x is an element of Set Nv and every digit of x is a '5')
Set N5 = { v5, v55, v555, v5555, v55555, ...}

Theorem: There is a bijection between N and Set N5

(Note: Sometimes I prove something that is obviously true. I usually do this to introduce the reader to the use of a critical concept or to set the stage for a future proof.)

Proof: Consider this proposed bijection between N and Set N5:

Element of N		Element of Set N5	Width
1	<-->	v5	[1]
2	<-->	v55	[2]
3	<-->	v555	[3]
4	<-->	v5555	[4]
etc.			

Note that the nth element of Set N5 has a width of [n]. To prove this is a bijection it will be shown that Set N5 maps into or onto N and that N maps into or onto Set N5. This would prove that the cardinality of Set N5 is less than or equal to N and vice versa, and would establish their equal cardinality (Schroeder-Bernstein Theorem).

Obviously, Set N5 maps into or onto N because Set N5 is a proper subset of N.

Suppose N could not map into or onto Set N5. If this were the case there must exist an element of N, say n, which does not map to an element of Set N5. Because of the diagonal construction of Set N5, the nth element has a width of [n]. Thus if some n, an element of N, does not map to an element of Set N5 then v555...555 (where the number of 5s represented by the three dots is n minus 6) must not be an element of Set N5.

This means 5555...555 is not an element of N (the 'v' was converted back to a '5'), and the cardinality of N is less than 5555...555. This means there does not exist any element of N with a width of [n+2], [n+3], [n+4], etc. But since n is an element of N, by assumption, then n+1 is an element of N and furthermore 10^(n+1) is an element of N. But 10^(n+1) has a width of [n+2]. This is a contradiction. **QED**

A critical thing to remember about Set N5 is that every one of its infinite number of elements has a **<u>different</u>** width. The importance of this will now be exploited. In fact, understanding the proof of this next theorem is absolutely critical and is the key to understanding the Creation Algorithm much later in the paper. The proof of this theorem also begins to explain the significance of the equal width of Set Nv and R(0,1).

5/9 Theorem: A sequence can exclusively use each element of Set N5 once and only once to construct 5/9.

Proof:

Let us define a sequence to use Set N5 to create 5/9:

{x **|** an ISD such that the digit in the nth DPI of x will equal the nth digit of the nth consecutive element of Set N5}

What happens in the above sequence can be graphically seen in this way:

(1) Element of N		(2) Element of Set N5	(3) DPI of 5/9	(4) Symbol	(5) Snapshot of 5/9
1	-->	v5	1	5/9[1]	.5
2	-->	v55	2	5/9[2]	.55
3	-->	v555	3	5/9[3]	.555
4	-->	v5555	4	5/9[4]	.5555
etc.					

The symbol "5/9[n]" means that if a snapshot of the sequence were taken after step n, this is what x would equal after that step. This snapshot is column #5 and is equal to the set of all FIS's of 5/9 (columns #1, #3 and #5); by the way the DT creates the NN.

1) It has already been demonstrated there exists a bijection between N and Set N5. Because every element of Set N5 has a different width every element of Set N5 contributes a digit to a different digit position of the sequence's proposed ISD.

Let us look at column #2 as a two-dimensional array of characters. This array has row numbers (column #1) and column numbers (i.e. the elements of its width set). Set N5, by its diagonal construction, has a bijection between its row numbers and column numbers. Thus the N that maps onto the rows and the N that is the width set of Set N5 are the same set (this is consistent with item #3c of the ACN). This means that if N, the row numbers, map onto the elements of Set N5, then N, the column numbers, map onto the digits contributed by Set N5 to the ISD.

Let the N from the column numbers be called Na and it maps from the width set of Set N5 to the DPI's of the proposed ISD. This means Na has a width orientation and includes every element of N.

2) By definition, N maps onto the digits of 5/9, meaning the set of all DPI's of 5/9 is N. Let this N be called Nb and it has a width orientation and is the width set of 5/9.

3) Both Na and Nb have a width orientation and follow all of the criteria of the ACN. By the ACN, Na and Nb must be the same set, therefore the digits created by the sequence and the digits of 5/9 are in bijection with each other and are the same set of digits,

4) Therefore the sequence's proposed ISD, x, and 5/9 are the same element. **QED**

The rule used in the sequence is a very simple rule that differs from the rule that creates the NN in one major way. The rule in the DT makes the nth digit of the NN **different** than the nth digit of the nth element of the list, whereas in the above sequence the nth digit of the ISD is **equal** to the nth digit of the nth element of Set N5.

Comments About Set R5:

Let Set R5 be the subset of Set Rt for which every digit of the element is equal to a '5':

Set R5 = {.5, .55, .555, .5555, ...}

Set N5 and Set R5 are obviously the same set, except that each element of Set N5 begins with a 'v' and each element of Set R5 begins with a decimal point: '.'.

Mathematicians sometimes state something to the effect:

"Set R5 is an infinite set of terminating decimals in R(0,1). 5/9 cannot be defined by a sequence that exclusively uses Set R5 because for each element of Set R5, following its significant digits there are an infinite number of zeros. Given any step n of the sequence the intermediate value of 5/9, called 5/9[n], has an infinite number of zeros. Thus, Set R5 cannot be used to define 5/9."

The answer to this question is very important because it gives us an introduction to the “Given Any” statement. The “Given Any” statement is a finite snapshot of an infinite sequence, and does not always capture the essence of the totality of the sequence.

I will give three of many answers to this statement:

Answer #1) Let us look at the sequence that builds the NN in the DT. Suppose the NN starts out, before the sequence begins, with an infinite number of zeros (i.e. the ISD = .0000...), and each zero is replaced by a '5' or '6' as the diagonalization sequence progresses. Given any step n the intermediate value of the NN, NN[n], has a finite number of significant digits, n, followed by an infinite number of unconverted zeros. By the above logic the NN never becomes a valid infinite expansion of R(0,1) and the DT fails to create an ISD and fails to prove R(0,1) is uncountable.

Answer #2) N maps onto the elements of Set R5, and the nth element of Set R5 has a width of [n] (counting only the 5s), meaning the nth element has a 5 in the nth digit position. We know the **set** Set R5 and the **element** 5/9 have the same width, counting only 5's, and subsequently the nth digit of the nth element of Set R5 can map to the nth digit position of 5/9 (i.e. we have an infinite set for which each element has a different significant width). Thus for every DPI of 5/9, Set R5 can contribute a 5 to that DPI's digit position. This accounts for all of the elements of N, meaning all of the DPI's of 5/9.

If, at the conclusion of the sequence, the ISD created by Set R5 contains both 5s and 0s, meaning it had an infinite number of zeros after all of the 5s which were contributed by Set R5; then which element of N, say n, maps to a digit position that contains a zero (i.e. which DPI contains a zero)? Such an element of N would be preceded by an infinite number of other elements of N, which is a contradiction to the way N is defined (i.e. no element of N can be preceded by an infinite number of other elements of N).

Or on the other hand, since every DPI is an element of N, if there was such an n, then Set R5 would be finite, with a cardinality of less than n, and therefore R(0,1) itself would be finite, with a cardinality less than 10^n.

Answer #3) Above, a sequence was designed which used Set N5 to generate 5/9. If this could not be done by Set R5, then for which element of N, say DPI number n, does 5/9 receive a '5' from Set N5 and not receive a '5' from Set R5?

In short, there would have to be two different definitions of "N" as column numbers in order for Set R5 to not be able to create 5/9.

The No Unique DPI Theorem:

No Unique DPI Theorem: No element of R(0,1) contains a digit position with a DPI, w, for which there are not 10^w elements of Set Nv which have a width of [w].

Proof: Every DPI of an infinite string is an element of N, by definition. Therefore w is an element of N. By the Nv Theorem there exist 10^w elements of Set Nv with a width of [w]. **QED**

Summary of Chapter III:

Sets of ISDs are looked at as two-dimensional arrays of characters. This is how the DT looks at R(0,1) because of its use of diagonalization, which requires a diagonal.

The width of a number is its set of DPIs. The width of a set is the union of the width sets of its elements. It has been shown that the width of a set can be transferred to the width of an element. Set N5 can be used to create 5/9. Much more will be said about "width" as this paper progresses.

Sets with a CPC have been introduced in order to later deal with the CPC of R(0,1).

Chapter IV: The Six Synchronized Uses of N and Linked Bijections

Looking at Set G in more Detail:

Set G was introduced above when the concept of "orientation" was introduced. If we apply diagonalization to Set G we can easily create one or an infinite number of NNs by using as many different algorithms as we wish to use.

Because Set G is listed as a SSA and as a set of ISDs, any reference to the digits of the NN or the columns of the array mean exactly the same thing. The NN is an ISD. Thus its digits are in bijection with the digits of any other ISD, and since the array is simply a collection of ISDs, all of the same width, then the digits of the NN and the columns of the array are the same thing.

Therefore, in the nth step of diagonalization the reference to the nth digit of the NN, and to the nth digit of the nth element, could instead refer to the nth column. For example, the DT could have stated: "the digit of the NN in the nth column and the digit in the nth column of the element in the nth row are different, thus the NN is not in the nth row, etc."

Diagonalization thus establishes a "link" between the nth column and the nth row because both the nth column and the nth row are mentioned in the nth step of the diagonalization algorithm.

Multiple Uses of Each n:

In the DT, N is used in six different ways because of the use of diagonalization. Not only is N used in six different ways, but each n in each N is synchronized. For example, for every step of the diagonalization sequence, say the nth step of the sequence; the same n refers six different things:

The Six Synchronized Uses of Each n in N:

1) The nth step of the diagonalization sequence, and
2) The nth digit of the NN (which is created in the nth step), and therefore,
3) The nth column of the array, and
4) The element in the nth row of the array, and
5) The nth digit of the ISD in the nth row of the array, and
6) n, an element of N, the same N which maps to the rows of Set G and the same N which is used to prove that the NN is not in row n (i.e. in step n, n is eliminated as being the row number which contains the NN).

Note that each of these items is equal to N, and that the same n in N is used in each of the items in step n of the algorithm. This means that each of these items is linked to the other. It is critical to note that these links apply to each and every step of the sequence. The reader should verify that all 6 of these items are valid for every element of N and for every step of the sequence.

The terms "linked," "simultaneous," and "synchronized" all mean the same thing, they mean that the same value of n is used in the same step. The term "synchronized" will take on added meaning later in the paper.

Given any two of the above items, there is a link between them in the diagonalization sequence. One of most important of the links is the link between the column numbers (item #3) and the rows of the array (item #4). This link is the basis for the concepts of "linked bijections" and "hinged sets." But this is not the only important link in the above list; another is the link between the digits of the NN (item #2) and the N that is mapping to the rows (item #6). This link will be the basis for the Set R1 Paradox mentioned near the end of this paper.

The Concept of Linked Bijection:

Consider this definition: "An SSA has a 'linked bijection' iff there exists a bijection between its column numbers and its rows (i.e. row numbers)."

Now let us consider Set G. Does Set G have a "linked bijection?" The answer is 'yes'. For every element of N, n, the nth step of diagonalization creates the nth digit of the NN, meaning the digit in the nth column. Also, the element in the nth row is compared to the NN and is eliminated from being the row containing the NN because its nth digit is different than the nth digit of the NN. Thus in the same step the nth column and the nth row are mentioned. This means there is a bijection built into diagonalization between the nth column (i.e. the nth digit of the NN) and (the element in) the nth row.

Diagonalization proves that Set G cannot contain all of the elements of R(0,1). Another way to say this is to say that if all of the elements of R(0,1) were put into an array, this array would not have a linked bijection.

This points out the two ways that the DT could have used Set G:

1) If an array has a linked bijection, then it cannot contain all of the elements of R(0,1), and
2) If an array contains all of the elements of R(0,1), then this array cannot have a linked bijection.

These two statements are contrapositives of each other.

More on the DOU:

It was mentioned above that the DOU and diagonalization are combined together because the DOU is not a mapping technique and the DOU must have a mapping technique and furthermore diagonalization needs a definition.

The DOU is a definition designed to be used with a very simple bijection between two sets, one of which is N. Diagonalization, on the other hand, synchronizes N in two different orientations of a SSA.

Because diagonalization is a synchronized, multi-orientation mapping technique, it must be used with a definition that interprets the results of the use of a synchronized, multi-orientation mapping technique! The DOU does not do that.

The DOU makes no reference to the column numbers of the array. The DOU treats the elements of R(0,1) as "single, unique symbols" (SUSs). In other words, the DOU treats each element of the listing as a single symbol, not a column of digits. The DOU in its pure form could be used on R(0,1) if the elements of R(0,1) were each treated as a unique, single Chinese-type symbol. Such a listing would have only 1 column! The DOU is an "infinite row, one column" definition.

Diagonalization, on the other hand, cannot work on a "one column" system of numbers. Diagonalization requires the set's array to have an infinite number of columns. Not even a finite number of columns would work for diagonalization.

The DOU, because it only references the rows of the array, and diagonalization, because it references both the rows and columns (i.e. the diagonal), are not compatible with each other.

Now consider this definition of "countable" and "uncountable."

Linked Bijection Definition (LBD): "An infinite set is countable iff its SSA has a linked bijection."

This definition interprets the use of a synchronized, multi-orientation mapping technique. The use of diagonalization converts the DOU into the LBD. The DOU contributes the 'iff' structure of the LBD and diagonalization contributes the "linked bijection" part of the definition.

The DT clearly uses the LBD rather than just the DOU for the reasons just mentioned: Set G has a linked bijection, thus when it is shown that the NN is not an element of Set G it is really saying that R(0,1) is uncountable because its elements cannot fit into an array with a linked bijection. Or it is uncountable because its complete listing cannot have a linked bijection. It is saying R(0,1) is countable iff its complete listing has a linked bijection.

That the DT uses the LBD is logical because diagonalization uses N and the column numbers simultaneously; meaning it could have used the column numbers instead of N.

The LBD leads to a LBDOC and a LBDOU. The key question now is this: is the LBDOU a valid definition of uncountable that is perfectly consistent with the DOC, Cantor's Countable Union Theorem (CUT) and other valid definitions and ways of proving a set is countable? Because of the use of the "iff" symbol and the use of diagonalization, the DT is actually assuming that the LBDOC, the LBDOU, the DOC, the DOU and the CUT are all perfectly consistent.

Summary of Chapter IV:

The “Six Synchronized Uses of N” have begun to be explored. Set G clearly is assumed to have a Linked Bijection. When it is shown that R(0,1) cannot be equal to Set G, it is really shown that a complete listing of R(0,1) elements cannot have a linked bijection. Because of this it is concluded that R(0,1) is uncountable. Thus it has been shown that the LBD is more consistent with the use of diagonalization than is the SD because the LBD deals with a synchronized, multi-orientation mapping technique.

Chapter V: Set Rj and the Logical Bubble Sort:

Set Rj:

The CUT proves that the union of a countable number of countable sets is countable. If we assume that R(0,1) is uncountable, we can take a countable number of disjoint, countable subsets of R(0,1) to create a single countable set which is not equal to R(0,1).

Let Set Rj equal the union of Set Rt and a countable number of disjoint subsets of Set Rnt. (the "j" stands for disjoint)

Set Rj is countable by construction.

Preliminary Ordering of Set Rj:

Let us consider this ordering of Set Rj:

1) Subset 1, all of the elements of Set R5, by size, and then
2) 5/9, and then
3) Subset 2, all of the elements of Set R6, by size (note: Set R6 is similar to Set R5 except that it uses '6's instead of '5's), and then
4) 6/9, and then
5) Subset 3, the rest of Set Rt in any order, and then
6) Subset 4, the rest of Set Rj in any order.

Let us now calculate the NN using the standard method:

List Number	Element of Set Rj	New Number
1	.5	.6
2	.55	.66
3	.555	.666
4	.5555	.6666
...		
none	5/9	completed
none	.6	completed
none	.66	completed
...		
and so on.		

Using this ordering of Set Rj, because Set R5 is in bijection with N it is quite clear that N will be completely consumed by Set R5 and no element of N will be available to map to 5/9ths or any element of Set R6, etc.

It is therefore quite clear that the NN is equal to .6666..., or in other words, the NN is equal to 6/9, an element of Set Rnt. The digits of the NN are not mapped to all of the elements of Set R5 union 5/9, much less all of the elements of Set Rt, and to none of the elements that come from Set Rnt.

In this ordering Set Rj does not have a Linked Bijection.

Before proceeding it is necessary to clarify some terminology that has already been used to a small degree.

The "Given Any Countable Subset" Concept Versus the "Complete Listing" Concepts:

Does the term "given any countable listing of a set" mean that the set **must** have a linked bijection, meaning its columns must be in bijection with its rows? Or does the term mean that the set is countable, whether the entire listing of the set has a linked bijection or not?

The DT obviously would side with the first of these concepts because Set G has a linked bijection. But this paper will side with the concept that any listing of any countable set is a valid "listing," whether the listing has a linked bijection or not.

This means that no matter how we order Set Rt, Set Rj or any other countable set, this ordering will be a valid "listing."

This means that when we refer to a "countable" set, or to a set assumed for demonstration purposes to be countable, we can automatically refer to any "listing" of this set as a "complete listing." This means that **all** of its elements are in the listing, whether it has a linked bijection or not.

The reader needs to be careful of this terminology. It is a new concept to think that a "listing" can include **all** of the elements of a set that is not ordered such that it has a linked bijection. When the DT talks about "given any countable subset" it is referring only to a set that has a linked bijection.

One purpose for this clarification is that the following two phrases mean the same thing:

1) "R(0,1) is uncountable because no listing of R(0,1) that has a linked bijection can contain all of the elements of R(0,1)" and

2) "R(0,1) is uncountable because a complete listing of all of its elements cannot have a linked bijection."

The DOU is designed to work with the first of these statements. However, if we assume for discussion purposes that R(0,1) is countable, we can reach the contradiction needed by the DOU by using the second statement.

Official Ordering #1 of Set Rj:

The above ordering of Set Rj was for demonstration purposes only. Now consider this official ordering of Set Rj:

First, Set Rt, first by width (i.e. elements of width [1], then elements of width [2], and so on); and within width by size. This is called the "native" order of Set Rt.

Second, the rest of Set Rj in any order (all of the rest of the elements of Set Rj are elements of Set Rnt, so their order is not critical to this example).

We will create the NN using the standard algorithm:

N	Set Rj	NN
1	.0	.5
2	.1	.55
3	.2	.555
4	.3	.5555
5	.4	.55555

and so on.

It is doubtful that any mathematician would question that the above listing is a valid mapping between N and Set Rj because it is in a logical ordering and it includes every element of Set Rj. By "logical ordering" I mean it is listed by width, as a valid bijection between N and Set Rt would be logically constructed.

But there is a problem. Note that Set R5 is completely embedded (note: by “embedded” I mean that a proper subset of a set has its elements scattered within the listing) within Set Rt and note that Set R5 has remained in its native order by width. If the NN cannot escape Set R5 in the first preliminary ordering of Set Rj above, it certainly cannot escape Set Rt (i.e. Set R5) in this example, because it cannot escape Set R5 in either example!

This means that Set Rj is countable, it is in a very logical order, but yet it does not have a linked bijection. This is one reason I prefer the terminology that I use.

In this case a countable number of different algorithms could create every possible NN.

Official Ordering #2 of Set Rj:

Now let us continue to order Set Rj as above, but we will convert all of the elements of Set Rt to unique elements of Set Rnt. For example, consider .9873205, an element of Set Rt. We will convert this element to a unique element of Set Rnt and Set Rj that begins with the string "9873205," such as .987320533309283... We will call this subset of Set Rnt: Set Rtt. We will design Set Rtt so that it is disjoint with the rest of Set Rj.

Now Set Rj is a countable subset of Set Rnt, meaning every element of Set Rnt is a nonterminating decimal. Can N map onto all of the elements of Set Rnt in this ordering (i.e. the same ordering as above, but using Set Rtt instead of Set Rt)? Obviously not, Set Rtt will consume all of the elements of N because the subset of elements of Set Rtt created from Set R5 will consume all of the elements of N before all of the elements of Set Rj are mapped to.

Thus we have an example of a countable set of infinite expansions that does not have a linked bijection when placed in a logical ordering.

The GTE Ordering of Set Rt:

Let us return to Set Rt for a moment. Consider this definition:

Greater Than or Equal Ordering (GTE Order): "A subset of R(0,1) is in a GTE Order iff given any element of N, n, the nth element of the set has n or more significant digits (consecutive, terminating zeros are not considered significant in this context), meaning a width of [n] or greater."

In this case R(0,1) is looked at as the union of Set Rt and Set Rnt. Let us consider a GTE Ordering for all of Set Rt:

```
N      Set Rt

1      .987089790812387  (15 significant digits, needs at least 1)
2      .42               ( 2 significant digits, needs at least 2)
3      .65423898127      (11 significant digits, needs at least 3)
4      .87251            ( 5 significant digits, needs at least 4)
5      .23489736         ( 8 significant digits, needs at least 5)
...
Undef  .173
...
```

Given this GTE Ordering of Set Rt, does any element of N map to .173? Since .173 has 3 significant digits, it can only be the 1st, 2nd or 3rd element in the above ordering, by definition. We can see by observation that it is not. Thus we conclude that .173 is not mapped to by any element of N. It is a NN.

Of course we can do this for an infinite number of other elements of Set Rt. We trivially conclude that if Set Rt is in a GTE Order there will not exist a successful **"Linked Bijection Mapping"** (LBM) between N and Set Rt, meaning the listing does not have a linked bijection. We conclude this because given any GTE Order listing of Set Rt we can easily find an infinite number of elements of Set Rt to which no element of N maps.

Now let us consider R(0,1). If we consider R(0,1) as a set of ISDs, then R(0,1) is **always** in a GTE Order, no matter what order it is listed in!

The Bubble Sort:

Most computer programmers are familiar with the "Bubble Sort." I will modify this technique slightly. Suppose we consider any listing of Set Rj. Now let use go through the following algorithm:

Step 1: We will find the first element of the list that has a '5' in its first digit position, say it is element m. Then we will swap or exchange the first element of the list with this element, which has a '5' in its first digit position, if necessary. The original mth element is now the first element and the original first element is now the mth element, if necessary. Either way the first element of the list has a '5' in its first digit position.

Step 2: We will start with the second element of the list and then find the first element we come to which has a '5' in its second digit position. We will then swap or exchange the second element of the list with this element, if necessary.

Step 3: We will start with the third element of the list and then find the first element we come to which has a '5' in its third digit position. We will then swap or exchange the third element of the list with this element, if necessary.

And so on.

The Bubble Sort does not change the cardinality of a set; it simply reorders it if necessary. It will be left to the reader to prove that for every n in N, this algorithm will be able to find an element with a '5' in its nth DPI as of the nth element or “later” in the listing. After doing the Bubble Sort the nth element of the list has a '5' in its nth digit position.

We know that all of the elements of N will be totally consumed by the subset of the list that has a '5' in the nth digit position. We also know that no element of N will map to any element of Set Rj that does not contain a '5' (Note: there will be many elements of Set Rj which do contain a '5' which are not mapped to by any element of N).

Thus, we have proven that after the Bubble Sort is executed Set Rj does not have a linked bijection. Is this a proof that Set Rj is uncountable? Obviously not.

The Bubble Sort is mentioned to introduce an even stronger concept.

The Logical Bubble Sort (LBS):

With the Bubble Sort now understood, let us now use a different method of placing a subset that consumes N at the beginning of a listing. In this case we will use R(0,1). We will call this the "Logical Bubble Sort" (LBS). Unlike the Bubble Sort, for the LBS we will need to see the order of the elements in the listing. Let us consider a listing of R(0,1) elements (we are assuming R(0,1) is countable and that every element of R(0,1) is in the listing):

```
N          R(0,1)

1   ->    .8903182...
2   ->    .1592983...
3   ->    .5991014...
4   ->    .3194836...
...
```

Now consider this algorithm, which is similar to the Bubble Sort:

Step 1: We will start with the first element of the list and then find the first element of the list with an '8' in its first digit position (note: '8' was chosen for the algorithm because it is the first digit of the first element of the listing). We will then swap or exchange the first element of the list with this element, if necessary (of course it is not necessary to swap elements because we have seen the listing and designed the algorithm so that the first element does not need to be moved).

Step 2: We will start with the second element of the list and then find the first element of the list with a '5' in its second digit position. We will then swap or exchange the second element of the list with this element, if necessary (ditto, it is not necessary).

Step 3: We will start with the third element of the list and then find the first element of the list with a '9' in its third digit position. We will then swap or exchange the third element of the list with this element, if necessary (ditto, it is not necessary).

And so on.

The LBS is a proof that no listing of all of the elements of R(0,1) can have a linked bijection.

Consider that we know: "any element of the list which does not have an '8' in its first digit position, and a '5' in its second digit position, and a '9' in its third digit position, etc. is not mapped to by any element of N.

We can easily find elements of R(0,1) which do not have an '8' in their first DPI, and do not have a '5' in their second DPI, and so on. The NN is an example of this but we do not need to use a formal diagonalization algorithm to prove this. It is obvious.

The LBS is designed to show how easy it is to find an infinite subset of R(0,1) at the beginning of any listing of R(0,1) that consumes N. If we were to assume or prove that R(0,1) is countable we would not be surprised to note that **first** the Bubble Sort would work just fine (as it does with Set Rt and Set Rj). We would also not be surprised that **second** the LBS would work just fine. These two techniques do exactly the same thing! The only difference is that with the LBS we see the listing first! The LBS works strictly with the properties of R(0,1), independent of its infinite cardinal number.

The LBS is different than diagonalization in the sense that diagonalization actually creates a specific NN that is not part of the linked bijection portion of the listing. The LBS simply states that many elements obviously exist that are not in the linked bijection portion of the listing.

What diagonalization and the LBS both do is identify an infinite set, the same set by the way, which has a linked bijection at the beginning of the listing.

Bifurcation of the DT:

With these things in mind we can bifurcate the DT (note: bifurcate is a legal term which means to divide a trial into two disjoint trials, or in this case, two disjoint processes). The DT only needs to prove that the complete R(0,1) listing does not have a linked bijection. It was just shown by using the LBS that the complete R(0,1) listing does not have a linked bijection, but no specific NN was identified.

The DT did not need to actually identify a specific element of R(0,1) to which no element of N mapped, rather it simply needed to prove that N and R(0,1) could not be placed into bijection. It was nice to identify the NN, but it was not necessary. Thus the DT can be bifurcated into two parts:

1) (Necessary) Prove that no listing of R(0,1) elements can have a linked bijection (i.e. prove that the complete R(0,1) listing cannot be mapped to by N, the set of column numbers), and
2) (Optional) Identify one of the elements to which no element of N maps.

The concept of bifurcation is very important to mathematics because R(0,1) is the only set which meets these three requirements:

1) It is an infinite set of ISDs, and
2) It is base 2 or above, and
3) It contains every possible permutation.

The last of these three items means that the NN is assured to be a specific element that is not in the linked bijection portion of the listing. **No other set has that luxury!** Thus to really understand whether the DT is valid it is absolutely necessary to deal with sets that do not contain every possible permutation! This means other techniques and logical definitions must be developed that work with non-CPC sets of infinite expansions.

These techniques must be able to prove the lack of a linked bijection on sets with a non-CPC. This requires bifurcation. The key technique that will accomplish this is "hinged sets," which will be developed as the paper progresses.

Summary of Chapter V:

Set Rj is countable by the CUT, but even when it is placed in a logical ordering it does not have a linked bijection. GTE Orderings, Bubble Sorts and Logical Bubble Sorts have all been introduced. All of these things show how easy it is for N to be consumed by a subset of a set before all of its elements have been mapped to.

Chapter VI: Double Diagonalization and the Assumption of Diagonalization

Set Rv:

Before getting into the arena of hinged sets it is necessary to have some sort of transition between linked bijections and hinged sets. This section will provide that transition and will make the concept of hinged sets easier to understand. It will also introduce the "Assumption of Diagonalization."

Definition: "Set Rv" - Set Rv is created by converting the initial 'v' of every element of Set Nv to a '**.**' (decimal point). Examples:

Set Nv	Set Rv	Width of Set Rv Element
v90210	.90210	[5]
v89000	.89000	[5]
v00010	.00010	[5]

Note that Set Rv and Set Rt are identical sets except that consecutive, terminating zeros are defined to be significant digits in Set Rv, but are not significant digits in Set Rt. Set Rv is countable because it is in bijection with Set Nv, by definition. Note that Set Rv and Set Nv both have a CPC, but Set Rt does not have a CPC. The Nv Theorem obviously applies to Set Rv also.

Set Rvnt:

Let us create a set called Set Rvnt, a subset of Set Rnt. To create this set we will first take the elements of Set Rv and for each of these elements we will select a single nonterminating decimal from Set Rnt that begins with the digits of this element of Set Rv. In other words, we will take each element of Set Rv and "expand" it into an element of Set Rnt. This is how we created Set Rtt from Set Rt.

For example, consider this element of Set Rv: .8960. From the four digits of this element we will select an element of Set Rnt, which begins with the string "8960," such as .8960'983674523..., and place this element in Set Rvnt. The apostrophe was added to delineate which digits of the number match the parent element from Set Rv. The digits from the parent element of Set Rv are called the "base" of the element of Set Rvnt. The digits to the right of the base (i.e. to the right of the apostrophe) are called the "extension."

Let us add a rule to state that each element of Set Rvnt is unique. It is easy to create Set Rvnt so that each element is unique. Thus Set Rvnt contains exactly one and only one element for each element of Set Rv.

Assumption of Diagonalization:

One of the assumptions of the DT is that the NN is "made different" than every element of Set G. In other words, in order to justify that the NN is not in the listing, it is assumed that the NN is compared to, and made different than, every element of Set G. If the NN was not made different than every element of Set G, then the DT cannot prove that the NN is not in the listing.

If Set G has a linked bijection, then clearly the NN can be compared to, and made different than, every element of Set G. However, Set G is supposed to represent any countable set. In other words, the LBDOU assumes that every countable set has a linked bijection.

According to the LBDOU we should be able to take any countable set and prove it has a linked bijection. This means that we can assume that for any countable set there exists a mapping from the column numbers of its array **onto** the row numbers of its array, since they are assumed to be in bijection. To represent the column numbers of its array we can obviously substitute the digit positions or digits of any ISD, such as the NN.

Given any countable subset of R(0,1), we can create a NN. While creating this NN we can simultaneously determine if its digits actually map to all of the rows of the array. This would determine whether the NN is compared to every element of the listing, and thus whether it is actually guaranteed to be different than every element in the listing, meaning it is guaranteed to not be in the listing.

In this case the values of the digits of the NN would be irrelevant because our sole interest would be in determining whether its digit **positions** can map onto the rows of the countable set.

Let us consider the Assumption of Diagonalization (AOD):

Assumption of Diagonalization (AOD): Given any countable character array of ISDs, there exists a mapping from the column numbers of the array **onto** the rows or row numbers of the array. This means that an ISD, such as the NN, can be created to be different than every element of any countable set.

While it is true that given any infinite set of ISDs, a NN can be created, the AOD goes further than this by stating that the NN can be made different than **every** element of any countable set. Since the same N that maps onto the rows of the array is treated as the same N that represents the DPIs of the NN, the AOD is very similar to the LBDOU.

Is the AOD True?

To see if the AOD is true we will use Set Rvnt. Given any ordering of Set Rvnt clearly we can create a NN that is an element of the universal set, R(0,1). However, in this case our immediate interest is in determining if the NN is actually compared to every element of Set Rvnt. We don't even care if the NN is an element of Set Rnvt. Our sole interest is whether the digits of the NN can map **onto** all of the elements of Set Rvnt.

While the reader might jump to the conclusion that the NN is not an element of Set Rvnt, such a conclusion would be based on the assumption that its digits were mapped to and thus the NN is compared to every element of Set Rvnt. If the (digits of the) NN cannot be mapped onto all of the elements of Set Rvnt, then the NN **might** be an element of Set Rvnt!

Every element of Set Rvnt must be **eliminated** from consideration as being equal to the NN. The only elements that are eliminated are the elements that are mapped to. All of this will be better understood shortly.

To see if the NN is really compared to every element of Set Rvnt we will use a technique called "Double Diagonalization." Double Diagonalization (DD) uses a modification of Lamoreaux's Algorithm (LA) and these terms will be used interchangeably.

While diagonalization creates a single NN, DD will create an infinite number of NNs, called DDNNs, which stands for "Double Diagonalization New Numbers." For each step of diagonalization, a single digit is "appended" to the NN. DD also appends a single digit to each DDNN, but it does this twice for each DDNN in each step. Here is the algorithm:

Double Diagonalization (DD): if the nth digit of the nth element of the listing is a 0, 1, 2, 3, 4 or 5, we will take each DDNN that existed in the prior step and append a '7'. Likewise, we will also take these same DDNNs from the prior step and create a different DDNN by appending an '8'. If the nth digit of the nth element of the listing is a 6, 7, 8 or 9, we will take each DDNN from the prior step and append a '3'. Likewise, we will also take these same DDNNs from the prior step and create a different DDNN by appending a '4'. This algorithm takes each DDNN from step (n-1) and creates two DDNNs in step n. This "doubles" the number of DDNNs in each step.

Rather than complicate things and give an extensive definition and discussion of this algorithm, I will simply use it.

Applying Double Diagonalization:

Let us analyze diagonalization and DD at the same time. Since Set G in the DT is listed in a random ordering (i.e. there is no attempt to determine a special ordering of Set G), let us also list the elements of Set Rvnt in a random ordering:

```
N       Element of Set Rvnt

1      .9867534243'8763653...
2      .050986'12897376456...
3      .100028736'18723645...
4      .5555982'0009828763...
and so on.
```

First digit position of NN and first element of Set Rvnt: The first digit of the first element (in the above listing) of Set Rvnt is a '9'. The first digit of the NN is .5, using the standard algorithm. Applying LA to this same '9' we note that the NN has not been compared to .3, nor has it been compared to .4. The first two elements of Set DDNN (the set of DDNNs) are .3 and .4.

Second digit position of NN and second element of Set Rvnt: The second digit of the second element of Set Rvnt is a '5'. The NN at this point is .56 (i.e. .5 had the digit ‘6’ appended to it). Applying LA to this '5' we note that the NN, as of the first two steps, has not been compared to .37 (i.e. we appended a '7' to the .3), nor has it been compared to .38 (ditto), nor to .47, nor to .48. Note that we took each DDNN from the first step and created two DDNNs in this step. Set DDNN will now equal the four elements, .37, .38, .47 and .48 (i.e. .3 and .4 were each appended twice). .3 and .4 are no longer members of Set DDNN since they have been appended and thus replaced.

Third digit position of NN and third element of Set Rvnt: The third digit of the third element of Set Rvnt is a '0'. The NN is .565. Applying LA to this '0' we note that the NN, as of the first three steps, has not been compared to:

1) .377
2) .378
3) .387
4) .388
5) .477
6) .478
7) .487
8) .488

These are the eight elements of Set DDNN at this point.

And so on for all of the other digits of the NN.

Note that in the nth step of the double diagonalization algorithm (DD) that 2^n elements of R(0,1) are identified to which the NN has not yet been compared (as of the nth step). DD is identical to diagonalization except that it creates an infinite number of elements of R(0,1).

Note that the NN is not compared to any of the elements of Set DDNN, a subset of Set Rnt and R(0,1). This is by design. When the NN is created its purpose is to prove that the NN is not an element of the listing. When a DDNN is created, its purpose is to identify elements of R(0,1) to which the NN has not yet been compared. This double use of diagonalization is actually where the word "double" in the term "Double Diagonalization" comes from. The creation techniques of diagonalization and DD are the same, but the purpose for their creation is very different!

Analyzing Double Diagonalization and Set Rvnt:

In the above example we created elements of R(0,1) that were not compared to the NN. However, we did not determine whether any of the elements of Set DDNN were elements of Set Rvnt, which is also a subset of Set Rnt.

Now let us look at the DDNNs; which for every specific element of N are finite expansions, as being the bases of elements of Set Rvnt. For example, after the third digit of the NN is processed, by the Nv Theorem there are 2^3 elements of Set Rvnt that have the same bases as the finite expansions in Set DDNN. Suppose we had identified these 2^3 elements of Set Rvnt in the above algorithm. This means that after the third digit of the NN is processed, 2^3 elements of Set Rvnt can be identified that have not yet been mapped to by the DPIs of the NN.

This constitutes a change in the purpose of the DD algorithm. When DD creates elements of R(0,1), we are generally interested in determining whether these elements are elements of the listing, Set Rvnt in this case. But now our interest has changed. We are no longer interested in whether the DDNNs are elements of Set Rvnt; we are interested in whether the DPIs of the NN can map onto all of the rows of the Set Rvnt array. This changes things quite dramatically.

As we process the DPIs of the NN, the number of elements of Set Rvnt that are identified in this fashion diverges to infinity. For example, after the 10th digit of the NN is processed, 2^10 elements of Set Rvnt can be identified that have not yet been mapped to by the DPIs of the NN. After the googolplex DPI of the NN, 2^googolplex elements of Set Rvnt can be identified that have not yet been mapped to by the DPIs of the NN.

By the Nv Theorem, and the No Unique DPI Theorem, we can do this for every DPI of the NN, meaning for every step of this algorithm, since the algorithm is based on the DPIs of the NN. This is because this is a test to see if the digits of the ISD (the NN) can map onto the rows of the array. Furthermore, the DPIs of the NN and the DPIs of the elements of Set Rvnt are the same set (even when only the width of the bases are considered).

An algorithm is defined "for every element of N." Nothing happens "after N." Nothing happens "ulter N," meaning "on the other side of N." Clearly, for "every element of N," say n, we can identify 2^n elements of Set Rvnt that have not yet been mapped to by the DPIs of the NN.

How, then, is it possible for the DPIs of the NN to map to all of the elements of Set Rvnt? For which element of N, meaning for which step of the algorithm, step n, is it not possible to identify 2^n elements of Set Rvnt that are "later" in the listing? For which element of N does the “2^n” number start to decrease?

It is important to understand that this discussion is not an attempt to map N onto Set Rvnt. That can be done very easily. This discussion is an attempt to map the DPIs of the NN onto Set Rvnt! In doing this, the DPIs of the NN and the DPIs of the elements of Set Rvnt are kept in bijection and are synchronized, the way the DT does it.

By keeping the DPI bijection synchronized, meaning by keeping the DPIs of the NN and the DPIs of the elements of Set Rvnt linked together and **synchronized**, in bijection; it is clear that for the totality of N the number of elements of Set Rvnt that can be identified by the algorithm continues to grow. At no time does the number start to diminish.

The fact is that diagonalization forces the synchronization between the DPIs of the NN and the DPIs of the elements of Set Rvnt. This is because the nth digit of the NN is compared to the nth digit of the nth element of the listing in the nth step of the algorithm. This link and synchronization are welded into the heart of diagonalization.

But there is obviously another side to this story. Because the bases of the elements of Set Rvnt are finite expansions, Set Rvnt can be ordered in the same way that Set Nv is ordered (using the bases). Thus, given any element of Set Rvnt in this native order, the element of N that maps to this row can be identified, and thus the DPI of the NN that maps to this row can be identified.

We now have quite a contradiction. "Given Any Element of N," there are 2^n elements of Set Rvnt that have not yet been compared to the NN as of step n. "Given Any Element of Set Rvnt," the column that maps to it can be identified. This is quite a dilemma. Which "Given Any Element of" statement is right?

All of these dilemmas will be resolved as the paper progresses and much more powerful tools are developed.

A Return to the Logic of the DT:

As with the DOU/diagonalization combination, the DT assumes the AOD is true without discussion.

If the AOD is assumed to be true, then clearly diagonalization proves that R(0,1) is uncountable.

On the hand, if R(0,1) is assumed to be countable, then double diagonalization applied to R(0,1) proves that the AOD is false.

As above, the DT only proves that both of these assumptions cannot be true.

Summary of Chapter VI:

The section does not contain any proofs. In fact I have raised more questions than answers. There are two major things I wanted to do with this section. First, I wanted to introduce the Assumption of Diagonalization. Second, I wanted to introduce the concept of linking, synchronizing, and keeping track of the bijection between the DPIs of the NN with the DPIs of the elements of Set Rvnt, all of which are infinite expansions.

If this bijection is kept track of, and the DPIs are synchronized, then clearly the DPIs of the NN have a considerable problem being compared to all of the elements of Set Rvnt, within the bounds of N. This synchronizing and linking of the DPIs is an introduction to the powerful concept of hinged sets.

Chapter VII: The Ladder Paradox and Hinged Sets:

Introduction of "Hinged Sets:"

While there is a strong relationship between "linked bijections" and "hinged sets," the concept of "hinged sets" is a quantum leap beyond the concept of "linked bijections."

Hinged sets take into account the concept of a relationship between two sets that are bound together by some common link, called a "hinge." In the discussion of the Ladder Paradox, the reader should pay close attention to how a single object, a clock in this case, causes a relationship to exist between two countable sets. This relationship first leads to a paradox, and when irrational numbers are mixed into these paradoxes, some interesting things happen.

Before dealing with infinite sets, let us first look at some finite sets to understand how the column numbers and row numbers of an array can be related to each other by formula.

Finite Examples of Linked Bijections:

Let us consider all of the elements of Set Nv that have a width of [100] and call this Set Nv_100. By the Nv Theorem this set has 10^100 elements.

If we list the elements of this set in a character array, in any order, this array has 100 columns and 10^100 rows. If we consider linked bijections for finite sets, then clearly Set Nv_100 does not have a linked bijection. We can even apply diagonalization to the listing and create a NN that has a width of [100] and is somewhere in the list between the 101st and 10^100th element.

In fact, for any n in N, we can create a similar set, Set Nv_n, which does not have a linked bijection because it has n columns and 10^n rows.

This establishes a formula to link the cardinality of the rows directly to the cardinality of the columns. Since this formula is valid for each element of N (i.e. for each possible width, n, of the set), there is clearly a link between the cardinality of the columns and the cardinality of the rows.

Let us call the difference between the cardinality of the columns and the cardinality of the rows the "Cumulative Gap." The formula for the Cumulative Gap for any Set Nv_n for any n in N is (10^n) minus n.

Now let us turn our attention to infinite sets and study the Cumulative Gap in the Ladder Paradox.

The Ladder Paradox:

Let us consider a ladder that has an inexhaustible length, meaning an inexhaustible number of rungs (i.e. rungs are added to the ladder as needed, thus there are never any leftover rungs).

Now consider two construction teams, the Red Team and the White Team. The Red Team will place red rings around as many rungs of the ladder as they can. The White Team will place white rings around as many rungs of the ladder as they can.

Both teams will be completely subject to the same clock, a clock that ticks once a minute and once for every element of N. This single clock controls how much work each team performs. In other words, both the Red Team and the White Team must place their rings timed in conjunction with this single clock. There are no exceptions.

The Red Team is capable of placing one red ring on one rung of the ladder for each minute the clock runs. The White Team, on the other hand, improves its efficiency as time passes and in the nth minute of the clock they can put 10^n white rings on the ladder, one on each of 10^n different rungs.

The clock begins. Consider the number of rings placed on the rungs of the ladder by each team for the first few minutes:

(1) Minute	(2) Red Team	(3) White Team	(4) Cumulative Gap (White Only)
1	1	10	10 - 1 = 9
2	1	100	110 - 2 = 108
3	1	1000	1110 - 3 = 1107
4	1	10000	11110 - 4 = 11106

and so on.

Let the clock tick once for every element of N. There is clearly a bijection between N and column #2; thus the red team places a countable number of rings. Also, remembering the Nv Theorem, Set Nv can map onto column #3 (e.g. Set Nv_3 can map onto the white rings placed in minute 3): thus the number of rings placed by the White Team is also countable.

Note that for every element of N in column #1 above there are "white only" rungs, column #4 - Cumulative Gap - meaning rungs that have only a "white" ring on them. In fact as the elements of N get bigger, the number of "white only" rungs on the ladder gets bigger and bigger. The limit of column #4 will diverge to infinity.

The cumulative cardinality of "white only" rungs increases by more than 1 element for each element of N, thus since there are a countable number of steps in the algorithm, the number of "white only" rungs is at least countable. All of this leads to the following conclusions at the end of the sequence:

1) A countable number of rungs will have red and white rings, and
2) A countable number of rungs will have only a white ring.

The single clock is a "hinge," meaning a single object that creates a relationship between two sets. Since both sets are "hinged" to the single clock, we must look at both sets relative to the single clock (i.e. from the viewpoint of the single clock). Since the two sets "grow" at separate rates, relative to the single hinge, there is the appearance that one of these sets is bigger than the other set. Of course, in this case it is trivial to note that these two sets are the same size.

I want to frame this discussion of hinged sets in three different settings:

Case 1) With the hinge in place,
Case 2) With the hinge removed, and
Case 3) Going back after the algorithm and looking for a separate mapping.

After discussing these three cases, I will then talk about the Cumulative Gap as a separate set of elements.

Case 1) With the Hinge in Place:

Nr (the set of red rings) and Nw (the set of white rings) are "hinged sets" as long as they are both permanently linked to the single clock or hinge. If Nr and Nw are hinged together, the red team can never "catch up" with the white team. It is the "gap" between the two teams that is placed into the Cumulative Gap. That is why it is called the Cumulative "Gap."

The "white only" rungs are actually a measurement of the different "growth" rates of the two sets **relative to** the single hinge. In other words, as we go down the listing the number of white rings grows more quickly than do the number of red rings. If this was a "race," the white team would easily win the race.

The "single clock" acts as a witness. As long as it operates, the single clock observes that the white rings are placed onto the rungs of the ladder faster than the red rings. When the clock "stops" operating it "closes its eyes" and both teams simultaneously stop placing rings. At the instant the clock closes its eyes it notices that there are far more white rings that have been placed than red rings. The clock makes a note of this fact and then ceases to exist thinking that Nw is "bigger" than Nr.

Talking about a difference in growth rates or a race makes little sense unless one is talking about hinged sets.

As another example, suppose two spaceships leave the same point at the same time, but one of them is traveling at the speed of light and the other one is traveling at 1 mile per year. Suppose a witness; with a clock in hand, simultaneously observes the two spaceships for every element of N (i.e. one element of N per minute). Will the slow spaceship ever "catch up" with the fast spaceship? Obviously not, if both of them travel synchronized to a single clock. When the witness stops the clock and both spaceships simultaneously stop, the fast spaceship is far beyond the slow spaceship and the witness thinks the fast spaceship has traveled much further than the slow spaceship.

As another example, if you put 10^n minus 1 apples in a bucket for each and every element of N, would you expect to find out at the end of the sequence that the bucket is empty? This is the Apple Paradox. If the bucket is empty, were did the apples go?

The apples represent the different growth rates and as long as the hinge is in place, meaning as long as the single clock ticks, the bucket does have apples in it! When the clock "stops" the bucket contains apples. But after the algorithm is executed, no apple can be identified that is in the bucket. This will be discussed in a moment.

A "hinge" causes a paradox. The paradox is that for every element of N the Cumulative Gap has elements in it, but in fact all of the sets I have talked about are very similar. Two sets can appear to have different cardinalities, but in fact they have the same cardinal number. The difference in growth rates, and the relationship caused by the hinge, creates these paradoxes.

Let us call Nr "short N" and let us call Nw "long N." They are both equal to N. The single hinge, and the growth rates of the two sets relative to the hinge causes the concepts of "long N" and "short N". They are both the same N, but they are not in bijection as long as the hinge is in place. A "hinge" causes the **appearance** that the Cumulative Gap exists.

It is only by removing the hinge or going back after the algorithm is over that the Cumulative Gap can be shown to be the null set because for every step of the algorithm, and at the instant the algorithm stops, the Cumulative Gap has elements in it!

Case 2) With the Hinge Removed:

What if we don't use a clock, but instead suppose we use two hand-held counters and use two judges, one for each team. Time is no longer a factor. In this case each judge simply presses their counter each time they see a ring placed. Both judges see a countable number of rings placed, in bijection with N. Each counter is clicked once for each element of N. Both counters are clicked the same number of times.

In this case there is no appearance of a Cumulative Gap because there is no such thing as "time" or "growth rates" or "hinges." Each time a ring is placed the counter is punched. Each judge sees a countable number of rings placed on his or her ladder.

Case 3) Going Back After the Algorithm and Looking for a Separate Mapping:

Once the algorithm is over and the clock stops, there are a countable number of "white only" rungs. What if another person wants to count the total number of rings placed by the "white team" after the contest is over? In this case N can be recalibrated (from being equal to Nr) and can easily be mapped onto the rungs that have white rings on them. This recalibration ignores the hinge, it ignores Nr, it simply starts over.

It is really the number of "white only" rungs that determines the total cardinality of Nw because it is obvious that the number of "red and white" rungs is countable. If the number of "white only" rungs that appear to exist is countable, so is Nw. If the number of "white only" rungs is uncountable, so is Nw. In the case of the Ladder Paradox the number of "white only" rungs is easy to calculate because it is easy to calculate the cardinality of Nw. Since Nw is countable, so is the number of "white only" rungs that appear to exist.

But this is a simple case. In more complicated situations there may be no ability to go back and recount the "white" rings. In may not be known whether there are a countable or an uncountable number of "white only" rungs.

A Discussion of the Cumulative Gap as a Separate Set:

Do elements of Nw exist that are not elements of Nr? In this case the obvious answer is 'no'. Given any element of Nw, say n, there exists a "minute m" that this ring was placed. Since m and n are elements of N or Nw, there must be a "minute m" and a "minute n" and there will be a red ring placed in minute m and minute n.

In this case, if the Cumulative Gap is considered as a separate set, it has no elements. The Apple Paradox applies because of the hinge.

To be more specific, let us assume an element of Nw, say nw, is an element of the Cumulative Gap. nw therefore maps to a "white only" rung. Since there is only one definition of N, then nw must also be an element of Nr. If nw is an element of Nr, nw must map to the red ring that was placed in minute nw (note: the white ring that was placed on this rung was placed much earlier than the red ring). nw therefore maps to a "red and white" rung and by assumption to a "white only" rung. This is a paradox (i.e. the Ladder Paradox) because a single element of N cannot map to both a "red and white" rung and a "white only" rung because these rungs are in different places on the ladder.

Furthermore, if nw did map to a "white only" rung, nw would be preceded by an infinite number of other elements of N, namely the elements of N that map to "red and white" rungs. But no element of N is supposed to be preceded by an infinite number of other elements of N, by definition (i.e. every n is preceded by n-1 elements of N, but n is finite, thus n-1 represents a finite cardinality).

The Cumulative Gap, while it appears to exist, cannot in reality exist because that would violate well known rules of mathematics.

In summary, the Cumulative Gap exists as a separate set as long as the hinge is in place and as long as the clock is running and when the clock "stops." But after the clock "stops," it is impossible to identify any of these "white only" elements of Nw.

This is the essence of the Apple Paradox. The Cumulative Gap exists as a set when the clock stops, but no elements can be identified that are in the Cumulative Gap. Such is the beginning of the paradoxes of hinged sets.

So we have a set that has elements and does not have elements. When the clock stops it has elements, but then when we look for the elements after the clock stops we cannot find any.

A Formal Definition of "Unsynchronized":

Because Nw and Nr grow at separate rates, and because they are related to each other by the singe hinge, these sets are defined to be “unsynchronized” or “not synchronized.” This means their growth rates are significantly different and they are hinged sets.

In the Ladder Paradox, Nr and Nw are equal but not synchronized. And therein lies a critical phrase: "equal but not synchronized" which means two sets are equal in cardinal number but they are hinged and do not grow at the same rate.

The concept of "hinged sets" is a new concept. This means there are no formal tools to use this technology in proofs. Because the concept is a valid concept and explains many paradoxes, as will be shown, some type of definition needs to be developed for use in proofs. This formalization of the concept of "hinged sets" will be a definition of "unsynchronized."

Definition of Unsynchronized Hinged Sets: "Given two infinite hinged sets, A and B, where A and B are both infinite and are logically hinged (i.e. joined) by a single hinge; then if the elements of N are synchronized with the steps of an algorithm which creates both A and B, and for every element of N there is exactly one element of A created, and for every element of this same N the growth of B includes as a minimum an exponential factor which essentially represents an exponential growth rate of B for every element of N (i.e. every algorithm step), then A and B cannot be synchronized, meaning they are unsynchronized.

Note how N, the algorithm steps, and the growth of A and B are all linked and hinged together. This means that elements for B are not defined **except** when an element of A is defined, and vice versa. A and B are simultaneously created, but they grow at separate rates. Because of the hinge A can never "catch up" with B.

Note also that there is nothing said about whether B is countable or uncountable. Certainly an uncountable set is unsynchronized with a countable set, but two countable sets can also be unsynchronized as has been shown.

An uncountable set will obviously contain actual elements in the Cumulative Gap, but a countable set will always appear to contain elements. Whether it does contain elements or not will be discussed in a moment. For now it is sufficient to note that the definition of unsynchronized does not distinguish between whether B is countable or uncountable.

The definition of unsynchronized hinged sets says nothing about growth rates that are greater than zero but are less than "essentially" exponential. Such a case will be ignored in this paper, and all hinged sets will be considered synchronized or they will be considered unsynchronized by at least an exponential differential growth rate.

It is critical to note that A and B are hinged sets that are logically related. A and B are not just any 2 sets, but for the totality of N (i.e. the algorithm steps) their growth rates are related and joined by a single hinge. The hinge allows the Cumulative Gap to be kept track of.

Is the Cumulative Gap unsynchronized in the Ladder Paradox? The Cumulative Gap grows at what I call a "cumulative exponential minus linear" growth rate (i.e. (10^1+10^2+...+10^n)-n), which is actually faster than a pure cumulative exponential growth rate (i.e. 10^n). Thus Nr and Nw are unsynchronized by the above definition.

The purpose of the Ladder Paradox is to provide an understanding of hinged sets relative to sets of numbers. Before expanding our discussion of sets of numbers a word of warning is needed.

Sets can be represented and viewed in many different ways. For example, Set Nv_1000 (i.e. the set of all elements of Set Nv which have a width of [1000]) does not have a linked bijection and is unsynchronized (using this term for finite sets as an example) because its 1000 columns cannot map onto its 10^1000 rows. However, if we represent this same set in base 1 (i.e. v1, v11, v111, v1111, ...), then its rows and columns will be synchronized because its width will be [10^1000].

Synchronized or unsynchronized hinged sets are determined using one representation at a time, meaning a set can be synchronized using one representation, but not be synchronized using another representation.

Also, whether a set is synchronized or unsynchronized is a function of an algorithm that can create the elements of the set. It has nothing to do with the order of the set. For example, Set N5 is synchronized, no matter what order it is placed in. It is synchronized even if it is in an order that does not have a linked bijection.

However, having said that, it is assumed that any set that can be created as being synchronized can also be ordered so that it is synchronized, and vice versa. Also, it will be assumed that if a set cannot be ordered so that it is synchronized, then it cannot be created so that it is synchronized. It will be proven below that if a set is created to be unsynchronized, it cannot be ordered to be synchronized.

Summary of Chapter VII:

When two sets are hinged together, and grow at exponentially different rates for the totality of the algorithm, they are defined to be "unsynchronized." This means that one of these sets appears to be bigger than the other set but in fact they may be the same size (i.e. have the same cardinal number). As long as the single hinge is in place "short N" can never "catch up" with "long N."

Chapter VIII: Creating Irrational Numbers Using Hinged Sets

Using Irrational Numbers to Represent the Red Rings:

Rather than let the elements of N be mapped or attached to each red ring that is placed, let us assign unique, random irrational numbers to each red ring that is placed, meaning we will map Nr onto a new set of irrational numbers. In other words, let us do these things:

1) During the Ladder Paradox algorithm we will assign a unique, random irrational number to each red ring as it is placed (i.e. or to the rung a red ring is placed on), and
2) As the red rings are placed we will simultaneously start the creation of some "other" irrational numbers (using something similar to DD); such that as irrational numbers are assigned to the red rings; none of the "other" irrational numbers being created is ever equal to any of the irrational numbers assigned to the red rings (we will use the elements of Nr as the DPIs of these "other" numbers), and
3) When finished, we will assign these "other" irrational numbers, called NNs, to the “white only” rungs in the Cumulative Gap.

Since there is a bijection between the red rings and the digits of any of the NNs, it is clear that no element of Nr will map to any of the "new numbers" (the Set R1 Paradox will prove this later in the paper). Thus each NN that is created is not attached to a "red and white" rung.

Once we know what these NNs are we can assign them to the "white only" rungs in the Cumulative Gap. This set of NNs is countable and they are assigned to all of the "white only" rungs.

We have now found a way to create actual elements for the Cumulative Gap. We don't have to worry about any of the elements of Nr mapping to these elements because the elements of Nr are the DPIs of these irrational NNs.

When we used Nw with the Ladder Paradox, we created a Cumulative Gap, however, when the algorithm was over we noted that multiple rules of mathematics were violated if we considered that the Cumulative Gap contained elements. In this case, however, no rules of mathematics are violated if we ignore Nw and assign irrational numbers to be elements of the Cumulative Gap.

If we ignore Nw, or "long N," and only consider the irrational numbers that are assigned to the various rings, the following two facts are evident:

1) The NNs are actual elements of the Cumulative Gap **relative to** "short N," and
2) The NNs are not actual elements of the Cumulative Gap **relative to** "long N."

The first of these statements is true because "short N" is consumed during the creation of the DPIs of the NNs, and thus does not map to any of the NNs. The second of these statements is true because Nw or "long N" can map onto all of the irrational numbers that are created.

The first statement above is highly relevant to the DT, as will be become obvious shortly. To deny the first statement as factual would be to deny that the DT creates the NN.

If we **only** use irrational numbers in the Ladder Paradox, and totally ignore Nw, there is no reason to be concerned if the "white only" irrational numbers are preceded by an infinite number of other irrational numbers. There are no rules to prohibit this; as there is with Nw.

We have actually found a way to mask the fact that the total number of irrational numbers assigned to white rings is countable. To this point the reader has no doubt assumed that if the total number of white rings is countable, there is no way to create elements for the Cumulative Gap. But there is.

This means we can **construct** identifiable irrational numbers and assign them to the "white only" rungs that belong to the Cumulative Gap, **even when** the total number of white only rings is countable!

We can do this because no element of Nr can possibly map to any of the elements of the Cumulative Gap, by construction. In other words, we used the elements of Nr as the DPIs of the NNs.

The important thing to note at this point is that we have ignored Nw or "long N." More will be said about "long N" in a moment.

Set Rvnt and Hinged Sets:

Let us create the elements of Set Rvnt. First, we will create the elements of Set Rvnt that have a width of [1], meaning the width of their base is [1]. The cardinality of this set is 10^1. Next, we will create the elements of Set Rvnt that have a width of [2] (ditto). Now the cardinality of Set Rvnt is 10^1 + 10^2. And so on.

Once we create the Set Rvnt array, we note that its rows grow similarly to Nw and its columns grow similarly to Nr. As we create Set Rvnt, we note that its rows grow more than exponentially faster than its columns (using the last DPI of their base widths to create the column numbers and also as the sequence steps).

The rows and columns of Set Rvnt are hinged sets because they are two different orientations or views of a single set, meaning the same set. The growth of the columns and the growth of the rows are hinged together, one new column, say n, for 10^n new rows. This is the very definition of a hinged set and of two sets being unsynchronized (in this case the sets are the set of column numbers and the set of rows)!

When using DD on Set Rvnt, we must keep in mind that there are two synchronizations going on at the same time. First, is a "width-to-width" synchronization. This is a required synchronization because the DPIs of the NN and the DPIs of the elements of Set Rvnt are the exact same set and must be kept in bijection and synchronization for every step of the algorithm.

The other synchronization is a "width-to-length" synchronization, meaning a "column-to-row" synchronization. In this case Set Rvnt is clearly unsynchronized, meaning a "width-to-length" synchronization attempt will fail.

Applying DD to Set Rvnt, however, has some interesting twists. First of all, because the "width-to-width" synchronization must be kept track of, and because two orientations are being considered simultaneously, the "hinge" absolutely cannot be broken! This means that Case 2 and Case 3 above do not apply! We cannot remove the hinge and we cannot go back and use "long N" because the use of "long N" would destroy (i.e. it ignores) the "width-to-width" synchronization, which is built into the use of diagonalization!

Thus in the prior section we were justified in ignoring "long N" because in this case the use of "long N" would violate the whole purpose of diagonalization.

The whole purpose of diagonalization is to prove that the NN exists. If, after diagonalization is done, we "go back" and look for the NN using "long N," then the whole purpose of diagonalization will be invalidated. This is important to remember.

A Game of Keep-Away:

As with Set Rvnt, R(0,1) also has two hinged sets, its column numbers and rows. Because Set Rvnt is an embedded subset of R(0,1), it is obvious that R(0,1), even if countable, cannot be synchronized.

To help understand the significance of hinged sets with regards to R(0,1), let us create a new set called "Set Rvnt-Infinite," or to use the shorter name: "Set Rvi." Whether we assume R(0,1) is countable or uncountable, for each element of Set Rv we can create a countable number of elements of Set Rvi that have this string as their base. This is identical to the way Set Rvnt was created, except in this case for each element of Set Rv we will create an infinite number of elements of Set Rvi.

Constructing a Set Rvi is not difficult, but it is time-consuming to consider different cases, and will be left to the reader.

Now let us set up a contest between two people. Person 1 claims that the DPIs of the NN can map onto (i.e. eliminate) all of the elements of Set Rvi and thus can eliminate them from being equal to the NN. Person 2 claims that Person 1 cannot do that. So they begin a contest using the following CUT type listing (i.e. if Set Rvi were listed as a CUT array, this is the pattern the array cells would be mapped to):

```
N      Element of Set Rvi

1      .0'4039283764733653...
2      .0'5984756897372456...
3      .1'4908273615984736...
4      .2'9087362761568650...
5      .1'0081109323874676...
and so on.
```

DPI #1: Person 1 maps the first column to the first row and creates the first digit of the NN, a '5', which is different than the first digit of the first element of the listing. The NN equals .5 at this point. Person 2 notes that Set Rvi contains three infinite subsets of elements that have not had any of their elements mapped to by the first DPI of the NN, namely the elements that have .5'..., .7'..., and .8'... as their bases.

All of the elements of Set Rvi in these three infinite sets are **<u>later in the listing</u>**. Eventually all of these elements must be mapped to (i.e. eliminated) by the DPIs of the NN. After the first step Person 1 notes that **<u>none</u>** of the elements in these three sets have been mapped to by the first DPI of the NN.

DPI #2: Person 1 maps the second column to the second row and creates the second digit of the NN, a '6', which is different than the second digit of the second element of the listing. The NN equals .56 at this point. Person 2 notes that Set Rvi contains five infinite subsets of elements that have not had any of their elements mapped to by the first two DPIs of the NN, namely the elements that have .56'..., .77'..., .78'..., .87'... and .88'... as their bases.

All of the elements of Set Rvi in these five (i.e. 1 plus 2^2) infinite sets are **<u>later in the listing</u>**. Eventually all of these elements must be mapped to (i.e. eliminated) by the DPIs of the NN. After the second step Person 1 notes that **<u>none</u>** of the elements in these five sets have been mapped to by the first two DPIs of the NN.

And so on.

After the nth DPI is mapped to the nth row, there will be one plus 2^n infinite sets, such that every element of these infinite sets is later in the listing. By the Nv Theorem and the No Unique DPI Theorem, this formula is valid for every element of N.

Who will win the contest? Person 1 will certainly not win the contest. The "Increasing Gap" between the column numbers and the rows certainly diverges as the elements of N get larger. A sequence is only defined for the elements of N.

The point to this discussion is that even though Set Rvi is countable, and can be mapped to by a DSM, clearly the DPIs of the NN will fail to map onto all of the elements of Set Rvi, relative to "short N."

This is because a DSM ("long N") looks at Set Rvi as an element array, and ignores the DPIs of any of the elements in the array. Diagonalization, however, looks at Set Rvi as a character array, and thus forges a hinge between two different orientations of the same set and forces a synchronization between the DPIs of the NN and the DPIs of the elements of Set Rvi.

Completing the Cycle:

Let us formally clarify some of the above comments.

Statement X1: In the Ladder Paradox, if the elements of N are used as the rung numbers, then we cannot identify an element of N that is in the Cumulative Gap. If an element of Nw maps to an element of the Cumulative Gap, say nw, nw would also an element of Nr and it would be preceded by an infinite number of other elements of Nr.

Statement X2: If the elements of Set Rnt are used as the rung numbers, and if we ignore Nw, then we can create NNs and assign them to the Cumulative Gap (whether the total number of irrational numbers is countable or uncountable). This is done by using the elements of Nr as the DPIs of the NNs.

Now, clear your mind for a moment. Suppose a set "Rw" was made up of irrational numbers in R(0,1) and had no connection to any set of finite expansions or to any elements with finite bases. Further suppose that we did not know the cardinality of Rw because we could not find a way to map N onto Rw. Even though we didn't know the cardinal number of Rw, suppose we were able to prove that its array was unsynchronized. With this knowledge suppose someone created irrational NNs and assigned them to the Cumulative Gap.

We would clearly conclude that the Cumulative Gap consisted of real elements and that it was definitely not the null set. We would conclude Rw was uncountable by the CDOU. We would do this using "short N." Because we don't know of a mapping between N and Rw, we cannot use "long N." The uncountability of Rw would be assured.

Because there is no Nw to test the cardinality of Rw, we could not go back and determine the cardinality of the Cumulative Gap or Rw. Because there are no finite bases, it could not be proven that the Cumulative Gap is the null set.

But now suppose someone comes along and using a technique other than a mapping says to the community that Rw is countable. Immediately someone in the community would ask these two questions:

Question 1) If Rw is countable, by the DOU then some specific element of N (i.e. some column number of the array) must map to every element of Rw, even the NNs in the Cumulative Gap, and
Question 2) If some element of N, say n, maps to one of the NNs in the Cumulative Gap, then n is a column number and maps to the NN and a contraction is reached.

Note how Question 1 refers to a "long N" mapping and the DOU. It basically says that if a set is countable that someone could find a "long N" mapping.

Question 2 is more subtle. It implies that if the Cumulative Gap is the null set for "long N" that it is also the null set for "short N." It essentially converts the null Cumulative Gap for "long N" into a null Cumulative Gap for "short N."

It has already been shown that a Cumulative Gap can have elements even if the underlying set is countable. Thus the creation of the Cumulative Gap is not a proof that the set is uncountable.

The combination of the two questions basically states that because the Cumulative Gap is not the null set relative to "short N" that the set will be considered to be uncountable until such time as someone comes up with a "long N" mapping. At that time the "short N" mapping will be considered a failure.

That is dangerous logic because the proof that the Cumulative Gap exists relative to "short N" says absolutely nothing about the cardinality of a set.

The above two questions are bogus and are full of false assumptions! These two questions fail to take into account the real reason the Cumulative Gap exists in the first place: that two sets are hinged and "unsynchronized." If a set is countable and is very complex it is quite possible that a Cumulative Gap can appear to exist relative to "short N." But the existence of the Cumulative Gap relative to "short N" says nothing about the cardinality of a set. Thus the creation of a NN, by itself, says nothing about the cardinality of a set.

Introduction to the Significance of Hinged Sets with R(0,1):

The columns and rows of a set of ISDs represent hinged sets. Let us assume R(0,1) is countable. Suppose that R(0,1) can be ordered in a logical manner and that there is no reason to suppose that all of the elements of R(0,1) are not in this ordering.

Now let us apply diagonalization to this ordering. Will diagonalization work, meaning can we create a NN? Since Set Rvnt is an embedded subset of R(0,1), it is clear that the rows of R(0,1) grow much faster than do the columns.

It is also clear that there is an Increasing Gap between the columns and rows. Because of this Increasing Gap the NN can easily be created. However, the NN cannot eliminate every element of the listing from being equal to the NN because the DPIs of the NN fall further and further behind in the elimination process (e.g. DD and the assumption that R(0,1) is countable).

If I were to say that R(0,1) is countable, the community would ask me these two questions:

1) "If R(0,1) were countable, some element of N, say n, would map to the NN, which is in the Cumulative Gap, and
2) If n maps to the NN, then n must also be a column number (i.e. an element of “short N”), which contradicts that the NN is in the nth row."

These are the very two bogus questions I just talked about. They assume the DOU is true, meaning they assume that if a set is countable its Cumulative Gap relative to "long N" will be the null set. Furthermore, they assume that if the Cumulative Gap relative to "long N" is the null set, then it must be the null set relative to "short N." But diagonalization creates the Cumulative Gap (i.e. the NN) relative to "short N."

How about "long N?" The study of "long N" is really a study of the many different ways that R(0,1) can be ordered. A side-by-side mapping and an ordering are really the same thing.

The DT should have included the study of many different ordering and mapping techniques. But because the DT makes no attempt to study different orderings, it really cannot conclude anything about "long N." It can only talk about "short N."

To answer the question whether the Cumulative Gap actually exists for "long N" for R(0,1) is far more complex than might be thought at this point. The "Creation Algorithm," soon to be discussed, and the "Set R1 Paradox," which is discussed near the end of this paper, will actually be needed to answer the question about whether the Cumulative Gap actually exists with R(0,1), and more importantly, whether the existence of the Cumulative Gap is a proof that R is uncountable (i.e. is the DOU true).

Summary of Chapter VIII:

A critical thing to understand about hinged sets is that irrational numbers can be created by using “short N” as their DPIs, and these NNs can be assigned to the Cumulative Gap. This can allow a countable set to actually have elements in its Cumulative Gap. It should not be assumed that a set with a Cumulative Gap is uncountable. And questions should not be formulated that assume if the Cumulative Gap is the null set relative to "long N" that it is automatically the null set relative to "short N."

Chapter IX: Kehr's Theorems

Kehr's First Theorem:

It is now time to formally start proving things using the definition of "unsynchronized." In the prior section, it was shown that Nw and Nr were not synchronized. Now a more general proof using the concept of hinged sets will be given. In this theorem Nr means N(row). Do not confuse this with the Ladder Paradox, where Nr stood for N(red).

Kehr's First Theorem: Let G be a 2-dimensional grid, where each cell of the grid can be empty or can contain exactly one dot. Let Nr be the rows of this grid and let Nc be the columns of this grid. Nr and Nc are both equal to N. Let the dots in this grid be created such that there is exactly one dot placed in each row of this grid, meaning there is 1 dot in row n, for each n in Nr. Furthermore, let there be 10^n dots in column n, for each n in Nc (e.g. there are 10^23 dots in column 23, there are 10^503 dots in column 503, and so on). Nr and Nc are hinged sets because they are two different orientations of a single grid. No matter how the rows of this grid are ordered, Nr and Nc cannot be synchronized.

Proof:

In this theorem G is created to be unsynchronized. This proof will demonstrate that it cannot be ordered such that it is synchronized.

Let us assume that the grid is synchronized.

In order to keep track of the dots; let us tag or attach an element of Set Nv to each dot in the grid. By the Nv Theorem the elements of Set Nv of width [1] can be tagged to the dots in column 1, the elements of width [2] can be tagged to the dots in column 2, and so on. Since:

1) The width of the grid is N, by definition, and
2) The width of Set Nv is N, by an above proof, and
3) Both the columns of the grid and the columns of the Set Nv array have a width orientation;

Therefore it is clear by the ACN that there is a bijection between the width of Set Nv and the width of the grid. Thus by the Nv Theorem there is a bijection between the elements of Set Nv and the dots in the grid. This means we can tag each dot once and only once with a unique element of Set Nv and each element of Set Nv tags one and only one dot.

This means there are a countable number of rows in this grid, no matter how it is ordered. Thus Nr (N for the rows) and Nc (N for the columns) are both equal to N.

To prove the theorem we must contend with ANY ordering of Set Nv.

If we order the dots in the grid in the same order as Set Nv in its native order, as elements of N, then the Ladder Paradox tells us that the rows and columns of this array are unsynchronized. Thus, let us consider a more random ordering. Obviously, Nr and Nc are hinged and we are assuming Nr and Nc are synchronized:

Nr	Set Nv	Column of Dot (i.e. width of element)
1	v98673452	8
2	v10091092837864	14
3	v98728	5
4	v8	1
5	v0000326	7

and so on.

Now let us see if Nr and Nc can be synchronized:

1st element of Nc - we note that there are <u>at least</u> (10^1)-1 elements of Set Nv which have a width of [1] which have not yet been mapped to as of the first element of Nc. Note that there are actually 10 elements of width [1] which have not yet been mapped to because the first element, v98673452 is in column 8, but we are taking a worst case scenario. This means that there are <u>at least</u> (10^1)-1 rows of the grid that have not yet been mapped to as of the first element of Nc.

2nd element of Nc - we note that there are <u>at least</u> (10^1+10^2)-2 elements of Set Nv which have a width of [1] or [2] which have not yet been mapped to as of the first two elements of Nc. Note that there are actually 110 elements of width [1] or width [2] that have not yet been mapped to because the first two elements have widths of 8 and 14, respectively, but we are taking a worst case scenario. This means that there are <u>at least</u> (10^1+10^2)-2 rows of the grid which have not yet been mapped to as of the first two elements of Nc.

3rd element of Nc - we note that there are <u>at least</u> (10^1+10^2+10^3)-3 elements of Set Nv which have a width of [1] or [2] or [3] which have not yet been mapped to as of the first three elements of Nc. Note that there are actually 1110 elements of width [1] or [2] or [3] which have not yet been mapped to because the first three elements have widths of 8, 14 and 5, respectively, but we are taking a worst case scenario. This means that there are <u>at least</u> (10^1+10^2+10^3)-3 rows of the grid that have not yet been mapped to as of the first three elements of Nc.

and so on.

We see that the minimum identifiable number of grid dots which have not been mapped to as of some element of Nc is (10^1+10^2+10^3+...+10^n)-n, which is the "cumulative exponential minus linear" pattern growth rate. This is the same pattern that we saw in the "Cumulative Gap" column of the Ladder Paradox in its native order. Thus, the dots in the Cumulative Gap grow by more than an exponential growth rate. Thus, the criteria of the definition of unsynchronized are met and the order of the rows of the grid is irrelevant and Nc and Nr are unsynchronized. **QED**

Corollary: Set Nv, Set Rv and Set Rvnt cannot be synchronized

No proof is needed for this corollary because Set Nv was used in the proof of the above theorem and Set Nv, Set Rv and the bases of the elements of Set Rvnt are all the same set.

This first theorem and corollary point out the power of the concept of "hinged sets." Using standard mathematics if I said that Nr and Nc could not be synchronized, I would be asked to "identify" a specific element of Set Nv which is not mapped to by a column number of the Set Nv array. This means I would be asked to identify a specific element of the Cumulative Gap, a set that may or may not exist!

Kehr's Second Theorem:

This next theorem is more restrictive than the above theorem. While it is obviously true, I will give a formal proof of it.

Kehr's Second Theorem: Let G be a 2-dimensional grid, where each cell of the grid can be empty or can contain exactly one dot. Let Nr be the rows of this grid and let Nc be the columns of this grid. Nr and Nc are both equal to N. Let the dots in this grid be created such that there is exactly one dot placed in each row of this grid, meaning there is 1 dot in row n, for each n in Nr. Furthermore, let there be an infinite and countable number of dots in each column n, for each n in Nc. Nr and Nc are hinged sets as above. No matter how the rows of this grid are ordered, Nr and Nc cannot be synchronized.

Proof:

By the CUT, there are a countable number of rows, thus Nr is N. Nc is also N.

In this proof we will first be shown the ordering of the grid. We don't care what the ordering is, but it needs to be shown to us first. In a proof by contradiction, let us assume Nr and Nc are synchronized.

Now let us "tag" some of the dots in the first column, and we will "tag" some of the dots in the second column, and so on. In fact we will "tag" the first 10^1 dots in the first column as we come to them as we go down the listing. This is possible because we are first assuming that the two sets are synchronized; thus we are assuming every consecutive row is mapped to by a unique consecutive column.

Likewise, we will "tag" the first 10^2 dots in the second column as we come to them as we go down the listing. Note: we will **not** wait until all of the first column dots are tagged before we start looking for the dots in the second column, we will start looking for dots in the second column starting with the very first row. Similarly, we will "tag" the first 10^3 dots in the third column as we come to them, and so on.

We have no fear of not finding the required number of dots to tag because of our initial synchronization assumption. Also, because each element of N is a finite number, say n, thus 10^n is finite, but each column has an infinite number of dots in it.

Now we will prove that Nc and Nr cannot be synchronized, given any listing of the rows of the grid:

1st element of Nc - we note that there are at least (10^1)-1 **tagged** dots in the first column which have not yet been mapped to as of the first element of Nc.

The rest of this proof is trivial.   **QED**

Summary of Chapter IX:

We have now begun to generate the tools that will allow formal proofs. These tools formalize the abstract concepts of hinged sets and the definition of "unsynchronized."

Kehr’s First Theorem generalizes the Ladder Paradox. Kehr’s Second Theorem is more restrictive than the first theorem, but it uses the same technique as the first theorem in its proof.

Chapter X: The HSSD and a Formal Disproof of the DT

The Hinged Sets Standard Definition (HSSD):

Consider this definition of "countable."

Hinged Sets Standard Definition (HSSD): "An infinite set is countable iff its array columns and array rows are synchronized."

This definition states that if a character array is unsynchronized, it is an uncountable set. This definition obviously can be split out into an HSSDOC and an HSSDOU. The HSSDOC is obviously true, but does not apply to all countable sets, such as to Set Rvi.

Note that the original DOC in the SD does not make any reference to the column numbers of the R(0,1) array. Also, at the beginning of the DT when the CDOU is used, the CDOU states that if R(0,1) were countable, it could be placed into bijection with N. The CDOU does not make any reference to the column numbers of the R(0,1) array.

However, when diagonalization is used on this listing, diagonalization **<u>does</u>** make reference to the column numbers, meaning the same N is used as the algorithm steps, the column numbers, the digit positions of the NN, the N which is trying to map to the rows, etc.

Let us talk about Set G. With Set G, because of the use of diagonalization, it is clear that there is a linked bijection between Nc from the column numbers and Nr from the row numbers. It has already been shown that the LBDOU is consistent with the use of diagonalization.

However, the LBDOU says nothing about hinged sets, and clearly the rows and columns of the array are hinged sets. Furthermore, the LBDOU says nothing about whether these rows and columns are synchronized hinged sets.

In fact it is clear that Set G not only has a linked bijection, but that its columns and rows are hinged sets and are synchronized. The rows and columns of the Set G array "grow" at the same rate because they are in bijection relative to the hinge.

When the DT determines that Set G cannot contain all of the elements of R(0,1), it concludes that R(0,1) is uncountable. It has been shown that a set that is created to be unsynchronized cannot be ordered such that it is synchronized. Thus an unsynchronized array cannot "fit" into a synchronized template. Set G is a synchronized template.

R(0,1) is clearly an unsynchronized array and cannot therefore fit into a synchronized template. According to the HSSDOU R(0,1) is uncountable.

We know, however, that Set Rvnt is not synchronized and cannot therefore fit into a synchronized template. By the HSSDOU Set Rvnt is uncountable as is Set Rvi. However, both sets are well known to be in bijection with "long N." Thus it is trivial that the HSSDOU is false.

The DOU is a "long N," single orientation definition. The DOU says nothing about arrays with multiple orientations, synchronized hinged sets and "short N."

The HSSD is a "short N," multi-orientation, synchronized hinged sets definition.

Diagonalization is a "short N," multi-orientation, synchronized hinged sets technique.

Clearly the use of diagonalization in the DT requires a definition that is compatible with the use of diagonalization. The HSSD is that definition, not the DOU and not even the LBD, a weak cousin to the HSSD.

The HSSD is a quantum leap above the LBD because the columns and rows of the array are hinged sets. The LBD does not take into account the paradoxes associated with hinged sets. We can now see that the DOU is not even close to being a compatible definition with diagonalization. Not only does the DOU ignore the hinged sets, but it also ignores the columns of the array altogether. The "true" definition of "countable" and "uncountable" that must be used with diagonalization is the HSSD.

While the definition of "unsynchronized" is not directly used by the DT to declare that R(0,1) is uncountable, the fact that R(0,1) is unsynchronized and is a set of ISDs allows the NN to be created by "short N" and to be assigned to the Cumulative Gap.

Thus the fact that R(0,1) is unsynchronized has an indirect bearing on the conclusion of the DT that R(0,1) is uncountable.

The Formal Disproof of the Diagonalization Theorem:

Now for the formal proof that the DT does not prove that R(0,1) is uncountable:

Disproof of DT Theorem: The DT does not determine whether R(0,1) is countable or uncountable

Proof:

It is claimed that the DT proves that N cannot map onto all of the elements of R(0,1). The N that cannot map onto the rows of the complete R(0,1) array, however, is synchronized with the column numbers of the array. Furthermore, the rows and columns of the R(0,1) array are hinged sets, thus diagonalization tries to synchronize the R(0,1) array.

Knowing from above that Set Rvnt is countable and cannot be synchronized, and knowing that Set Rvnt is a proper subset of R(0,1), and knowing that Set Rvnt and R(0,1) obviously have the same width, then adding the "rest" of the elements of R(0,1) to Set Rvnt, to create a complete array of R(0,1) elements, will add to the number of rows of the Set Rvnt array, but will not add to the number of columns of the Set Rvnt array.

This means that the complete R(0,1) array is unsynchronized. In short, we know that its rows grow at least "cumulative exponential minus liner" faster than its columns, because Set Rvnt is an embedded subset of R(0,1).

Thus R(0,1) meets all of the criteria for the formal definition of unsynchronized by KFT.

Let us assume that R(0,1) is countable. Let us see if diagonalization will work.

As the DT does, we will synchronized the DPIs of the NN with the DPIs of the other elements of R(0,1). Furthermore, we will link the row numbers with the column numbers. We can thus use the column numbers as N.

Let us look at this listing of R(0,1) elements (note that N is Nc, and thus there is no relationship between this listing and "long N" or Nr or the DOU):

```
Nc      Element of R(0,1)

1       .98675342438763653...
2       .05098612897376456...
3       .10802873618723645...
4       .55559820009828763...
and so on.
```

Column Number #1: Because R(0,1) is unsynchronized, and because it is a list of ISDs, we will use the DPIs of the ISDs (i.e. the column numbers), to construct digits for irrational numbers for the Cumulative Gap. Proving that the Cumulative Gap contains elements relative to Nc (short N) is all we need to do because the DT does not deal with "long N."

Note that the first digit of the first element of the array is a '9'. By KFT and the fact that Set Rvnt is an embedded subset, we know that R(0,1) has at least a "cumulative exponential minus linear" "Increasing Gap" growth rate. Thus there exist at least 10^1 minus 1 elements of R(0,1) that we can put into the Cumulative Gap. Let us pick .5, .3, and .4. (Note: we could put more elements in, but that is not necessary). Note that the width of these elements is consistent with the width of the NN at this point, thus we are keeping the width-to-width synchronization intact.

Column Number #2: Note that the second digit of the second element of the array is a '5'. By KFT there exist at least 10^1 + 10^2 minus 2 elements of R(0,1) that we can put into the Cumulative Gap. Let us pick .56, .37, .38, .47 and .48.

Column Number #3: Note that the third digit of the third element of the array is an '8'. By KFT there exist at least 10^1 + 10^2 + 10^3 minus 3 elements of R(0,1) that we can put into the Cumulative Gap. Let us pick:

1) .565,
2) .373,
3) .374,
4) .383,
5) .384,
6) .473,
7) .474,
8) .483,
9) .484

And so on,

By KFT, and the fact that Set Rvnt is an embedded subset, in general we can put at least: (10^1+10^2+10^3+...+10^n)-n elements into the Cumulative Gap in each step. The above algorithm puts 1+(2^n) elements into the Cumulative Gap in each step. Namely we have put the NN and Set DDNN into the Cumulative Gap. Thus we have not put too many elements into the Cumulative Gap for any element of Nc.

This means that if R(0,1) were countable, KFT would still apply and we could still assign the NN and all of Set DDNN into the Cumulative Gap!

Thus, if R(0,1) were countable we could still create the NN and Set DDNN and they would be in the Cumulative Gap.

Furthermore, if R(0,1) were countable, the NN would not be compared to every element of R(0,1) (i.e. Set DDNN) and the AOD would be false! This means that even if R(0,1) were countable, the creation of the NN would not eliminate every element of R(0,1) from being equal to the NN.

In other words, there is nothing in the DT to distinguish between whether R(0,1) is countable or uncountable because R(0,1) is a set of ISDs and has unsynchronized hinged sets. The DT fails to consider "long N" and only assumes the validity of the DOU and AOD. **QED**

Mathematicians are accustomed to assuming that the NN is compared to every element of the listing. However, what I have shown above is that if R(0,1) were countable, and if all of the elements of R(0,1) were in the array, the NN could still be created, but it would not have been compared to every element of the listing. Thus **it cannot be claimed that the NN is not in the listing!** The rows "grow" much more quickly than do the columns, to which the digits of the NN are in bijection.

Now it is time to actually determine the cardinality of R. To do this, definitions will be produced that determine the cardinality of a set based on the algorithm that creates it. These definitions will not be subject to the paradoxes related to side-by-side mappings between sets of vastly different properties. Nor will they be subject to hinged sets, which are also a problem related to mappings between existing sets.

Summary of Chapter X:

The specific error in the DT is that the DOU and diagonalization are not compatible with each other. The HSSD is compatible, but it is obviously false. This means that the DT does not have a valid definition, consistent with diagonalization, to prove that R(0,1) is uncountable.

By theorem, it has been shown that diagonalization would work exactly the same whether R(0,1) were countable or uncountable. R(0,1) is unsynchronized because it has two infinite orientations that are hinged together and "grow" at vastly different rates. The disparity in growth rates has nothing to do with the cardinality of R(0,1). Many sets known to be countable have vastly different growth rates. This disparity in growth rates allows a NN to be created before it is compared to every row of the array, whether R(0,1) is countable or uncountable. The NN is an element of the Cumulative Gap relative to "short N," but that is not a proof that R(0,1) is uncountable. Sets known to be countable have valid Cumulative Gaps relative to "short N."

My discussions so far are not a proof that R(0,1) is countable, they are simply a proof that the DT does not prove that R(0,1) is uncountable.

Chapter XI: The Partition Definition and Partition Theorem

What Do Countable and Uncountable Really Mean?

By now it should be obvious that the HSSD is not a valid definition of "countable." The HSSDOC is too restrictive. In fact if the mathematical community only used the HSSDOC to prove sets are countable, then almost all sets would be uncountable. It is valid when it works, but it rarely works because it requires synchronized hinged sets. The HSSDOU is simply false. It can be used to prove almost any set is uncountable.

While a "pure" DOC might seem a way out, in fact it is virtually impossible to prove a set is countable with the pure DOC. It is hard to ignore the column numbers of an array.

But perhaps the most suppressing thing about determining the cardinality of a set is the creation of NNs. It will be shown below just how easy it is to create NNs, whether using diagonalization, "long N," or nested intervals. In fact it is hard not to be able to create NNs.

With all of this in mind, it is clear to me that other definitions must be sought after which are not vulnerable to all of the problems of hinged sets, and other problems to be discussed later in this paper.

When I took my first course in Set Theory in 1964 I was taught that "cardinality" was a concept of "size." I was taught that cardinality is an issue of "how big" a set is.

At the beginning of this paper is a very simple definition of countable. This definition is oriented to the "size" of the set, not to its mapping properties. It gives mathematicians great leeway in determining the "size" of a set and does not restrict them to using one or two specific types of mapping techniques.

Most textbooks use the SD or some derivation of the SD that uses the term "denumerable." I do not like the term "denumerable" because of its equivalence with the DOC and the HSSDOC.

By the use of the SD in the majority of textbooks there is an implicit assumption in mathematics that "mappings" are an infallible and the only technique allowed for determining "how big" a set is. Certainly, the <u>**existence**</u> of a mapping is sufficient proof that two infinite sets have the same cardinal number. However, these definitions also assume that the <u>**failure**</u> of a specific type of mapping between two specific sets of vastly different properties, and which are hinged, proves these sets have different cardinal numbers.

This paper will introduce three new definitions of "countable." All of these are "if" definitions, meaning meeting the requirements of these definitions is sufficient to prove that a set has the same "cardinal number" and "size" as N. All three of these definitions determine the cardinality of a set based on an algorithm that can create it.

Each of these three definitions are just as logical at creating a sufficient condition to determine the equal cardinal number of two sets as is a pure DOC. None of them attempts to set up a sufficient condition to prove a set is "uncountable," as the SD does with its DOU. In other words, none of these are "iff" definitions.

In studying these definitions note that these definitions, if their criteria are met, do not guarantee that a mapping will exist between N and another set! If fact just the opposite is true. These definitions are designed **because** specific mapping techniques have already been shown to fail between two sets of equal cardinal number.

In other words, these definitions are designed to be used in Transfinite Set Theory (TST) in order to avoid the problems of "hinged sets," and other problems.

The "bottom line" to all of this is that the cardinality of a set, meaning the set of definitions and axioms which are used in the system, must be **driven** by the concept of the "size" of the set and not by the failure of a specific mapping technique.

In other words, the term "uncountable" must mean that a set has a very different cardinal number or size than N. Since, by definition, N is the smallest infinite set, then any infinite set which does not have the same cardinal number as N must be "bigger" than N. Much bigger, in fact, because an uncountable set cannot be broken down into a countable number of countable subsets (i.e. the CUT).

The first of these three definitions will now be discussed.

The Partition Definition:

A "partition" of a set is a collection of proper and disjoint subsets of a set. Each subset must be disjoint; meaning no element of a set can appear in more than one subset. Also each subset must have at least one element.

Normally to say that a set is "partitioned" is to say that the union of all of the disjoint subsets is equal to the set itself. However, in the case of determining cardinality, the concept of "partition" can certainly be extended to a case where the union of all of the subsets is a proper subset of the original set. This is especially true if the proper subset is the portion of the set that has the unknown cardinal number.

The Partition Definition is a definition that defines the maximum number of disjoint subsets that can exist based on the algorithm that creates the subsets.

The Partition Definition (PD): "Let us take a finite number, c, and an infinite set, S, and a sequence and algorithm based on N (meaning for each consecutive element of N the algorithm is executed exactly once). Let the first algorithm step divide S into no more than c disjoint subsets. Let each subsequent and sequential algorithm step divide a single subset of S, one that was created in a previous step, into no more than c disjoint subsets. By definition this means that for each step the cumulative number of disjoint subsets does not increase by more than c additional disjoint subsets. If these rules are followed then the algorithm creates a partition of S that consists of a collection of a finite or countable number of disjoint subsets. The partition, or union of the disjoint subsets, may be a subset of S (e.g. if elements of the set drop out because of the algorithm) or all of S."

(**Very Important Note:** nothing in the PD states anything about whether S is countable or uncountable. Nor does the PD say anything about the cardinality of the elements of S in each disjoint subset. The PD only states that S is partitioned into a finite or countable number of disjoint subsets)

The Partition Definition is a definition designed to set up a "sufficient" condition to state that a set, or one of its proper subsets, is made up of the union of a finite or countable number of disjoint subsets.

Note that the cumulative number of disjoint subsets increases by no more than c additional subsets for each and every sequential operation. This means that given any step of the sequence, say the nth step, the cumulative number of subsets is finite (i.e. no more that c times n) and can therefore map **into** N. This is true for every element of N. The PD is consistent with induction as well as Cantor's CUT. Most importantly, the PD is consistent with common sense. The PD is just as logical as the DOC.

Using the Partition Definition:

Let us look at R(0,1) as a set of points on the real number line. Let "(0,1)" be a line (i.e. R(0,1)).

Definition: "Partition Point" A "partition point," p, is an element of R(0,1) that divides an open interval in (0,1), say interval (a,b), where a and b are elements of R(0,1) and where a is less than p and p is less than b, into two open intervals: (a,p) and (p,b).

The Partition Theorem: (0,1) has a countable number of points

Proof:

Let us take Set Rt and order it first by width (i.e. elements with a width of [1], then elements with a width of [2], ...) and within width by size (i.e. greater than):

N	Set Rt
1	.0
2	.1
3	.2
4	.3
5	.4

and so on

Now let us create a sequence on N. For each element of N we will make the nth element of Set Rt (from the above listing) a partition point of (0,1):

N	Set Rt	Cumulative # of Subsets	
1	.0	1	(0,1) - original open interval
2	.1	2	(0,.1) & (.1,1)
3	.2	3	(0,.1), (.1,.2) & (.2,1)
4	.3	4	(0,.1), (.1,.2), (.2,.3) & (.3,1)

and so on

N is clearly in bijection with Set Rt. This same N is the basis for the algorithm steps. In other words, the algorithm is defined and executed once for each consecutive element of Set Rt. Each element of Set Rt increases the cumulative number of subsets of R(0,1) by 1. Thus the criteria of the PD are met and this sequence divides (0,1) into a countable number of disjoint subsets.

If induction were used instead of the PD, then if induction fails there must exist some smallest element of N, say n, for which a single element of Set Rt would divide (0,1) into more than one additional subset. However, given any element of Set Rt it is trivial to determine exactly which previously created subset is divided into two subsets.

Note that because the elements of Set Rt are all partition points, by definition, that the resulting subsets contain nothing but elements of Set Rnt. In other words, the union of all of the resulting subsets is obviously equal to Set Rnt because Set Rnt is the compliment of Set Rt relative to (0,1). All of the elements of Set Rt "drop out" of the disjoint subsets.

Now let us consider any one of these countable subsets: Iv. Suppose Iv contains two elements of Set Rnt: r1 and r2. By definition, every element of Set Rt is a partition point, thus no element of Set Rt is an element of Iv, which is a subset of R(0,1). Thus, the interval (r1, r2) is an open interval within Iv and within (0,1); and further Iv contains no elements of Set Rt. This is a contradiction that Set Rt is dense in R(0,1). In other words, given any open interval in R(0,1), it is trivial to find an element of Set Rt that is an element of this open interval. Thus Iv contains exactly one point of Set Rnt.

We also know that Iv contains only a single point by applying Cantor's Nested Intersection Theorem or Nested Interval Theorem (NIT) to the algorithm. Given any element of Set Rnt, say nt, we can specifically identify an infinite number of open intervals generated by the above algorithm which meet the criteria of the NIT and thus have as their intersection only one element, namely nt.

Note also that some of these nested interval sequences converge to an element of Set Rt, and thus the intersection of all of these nested intervals is a null set. These are not valid subsets, by definition, and thus are ignored.

Thus every resulting subset contains a single element of Set Rnt. Because there are a countable number of subsets and every subset contains one element of Set Rnt, Set Rnt is countable. Thus R(0,1) is the union of two countable sets, Set Rt union Set Rnt, and by Cantor's Countable Union Theorem, R(0,1) is countable. **QED**

This proof can best be visually understood by looking at Set Rv (because it has a CPC) as a base 10 tree, where the branches expand from left to right (like a pedigree chart). Consider that “level n” of the tree represents the elements of Set Rv of width [n]. For example, level 100 contains the 10^100 elements of Set Rv of width [100]. The algorithm moves from level 1 to level 2, and so on. The gaps between the "branches" on each level are the subsets. Also there is a subset outside the top and bottom branch (because 0 and 1 are outside of the tree). Thus after level 100 is executed there are exactly (10^100)+1 subsets.

As the different levels of the tree are executed the number of subsets increases exponentially. By several theorems above there is a bijection between the width of Set Rv and the width of Set Rnt. The ultimate number of subsets is the limit of the process of going across all of the levels.

A limit is defined "for every element of N." In this case N is the width of the array. Nothing happens after all of the elements of N are executed. For every element of N the total number of gaps is directly tied to the total number of branches plus 1. The only way that the number of gaps can become uncountable is if the number of branches becomes uncountable because Set Rv and R(0,1) have the same width.

For Set Rnt to be uncountable the width of Set Rnt would have to be greater than the width of Set Rv, meaning the ACN and No Unique DPI Theorem would be false. But there is a bijection between the width of Set Rv and the width of Set Rnt.

Summary of Chapter XI:

The PD is a very short and simple proof that R(0,1) is countable. The PD itself is not only logical, but is consistent with induction. The proof is also consistent with the NIT.

Chapter XII: Diagonal Sets and The ISD Theorem:

Set FIS:

As was mentioned above, the NN can be created by a sequence that, for every element of N, creates one digit of the NN, and thus one FIS for each element of N. It is obvious that any ISD, such as the NN, can also be created by a sequence on N.

This means that every ISD can be defined by an infinite set of FIS's, such that the nth FIS has a width of [n], where all zeros are considered significant. This will be proven in a moment. But first:

Theorem: There is no FIS of any element of any ISD that is unique to that one ISD.

Proof:

Suppose there was such an FIS; call it f, which must have a finite width, say [n], which is unique to fi, an ISD. Since n is an element of N, by definition, this would mean that R(0,1) itself had a width of [n] (by the permutation nature of R(0,1)). This means that by the CPC of R(0,1) the cardinality of R(0,1) would be at most 10^n, which would be finite because n is finite. But R(0,1) is not finite. **QED**

It will be left to the reader to prove that each FIS is the FIS of an infinite number of different ISD's.

Definition: "Set FIS": every unique FIS of every element of R(0,1).

If a person were to take all of the FIS's of all of the ISD's in R(0,1), and then remove the redundant ones, what is left would be Set FIS.

The FIS Theorem:

FIS Theorem: For every element of N, n, there exist 10^n elements of Set FIS with a width of [n].

The proof of this will be left to the reader. Similar to the proof of the Nv Theorem, except the contradiction is that R(0,1) would be finite because its width would be less than or equal to [n], where n is an element of N.

Note that no matter how many elements of R(0,1) exist, the first FIS of each element must come from a pool of 10 FISs: {.0, .1, ..., .9}. Likewise the second FIS of every element of R(0,1) must come from a pool of 100 FISs: {.00, .01, ..., .99}. And so on.

Because every FIS is a finite expansion, by definition, it turns out that it is easy to prove Set FIS is countable:

Set FIS Cardinality Theorem: "Set FIS is countable"

Proof: In this proof a bijection between Set Nv and Set FIS will be shown, again using the two-directional "into or onto" logic.

Let us propose a mapping from Set Nv into or onto Set FIS. First it will be shown that: given any element of Set Nv, say element vf, if we remove the initial 'v' from element vf and replace it with a decimal point, and call it element f, then f is an element of Set FIS.

Suppose this were not an "into or onto" mapping. Then there exists an element of Set Nv for which there is no corresponding element of Set FIS. Suppose this element of Set Nv is v532...786, where the number of digits represented by the three dots is n-6 (i.e. n minus 6). Then .532...786 is not an element of Set FIS, by assumption. This means the width of Set FIS, and therefore the width of R(0,1), is [n] or less. This means R(0,1) is finite, which is a contradiction.

Similarly it can be shown that there is a mapping from Set FIS into or onto Set Nv, or the width of Set Nv would be [n] or less, making N finite. This means Set Nv and Set FIS are in bijection.

We know that Set Nv is countable because it is a proper subset of N and therefore Set FIS countable. **QED**

By now the reader should realize that, except for the cosmetic differences in their initial symbol, Set FIS and Set Nv are basically the same set. Set FIS and Set Rv **<u>are</u>** the same set, except that they have different origins.

This means that the Nv Theorem not only applies to Set Nv, but also to Set FIS.

If the Nv Theorem and No Unique DPI Theorems were false then a subset of Set FIS could not create an ISD. When the DT creates the NN it creates an infinite number of FIS's, which, when appended together, create a single ISD. There is no "gap" between the width of the set of FIS's and the width of the ISD. Nor is there a gap between the width of Set Nv and the width of Set FIS.

Definition of Diagonal Set:

Definition: A "diagonal set" is a subset of Set Nv (or Set Rv) which meets 3 requirements (note: when the elements of Set Rv are used to create a diagonal set the set will be a special case of a Cauchy Sequence):

a) the set is an infinite subset of Set Nv (or Set Rv), and
b) the nth element of the set has a width of [n], and
c) given the nth element of the set, where n is greater than 1,
 the string of the first n-1 digits of the nth element is
 equal to the n-1th element of the set.

Examples: {v6, v67, v679, v6790, v67902, v679024, ...}
{.1, .16, .163, .1639, .16390, .163901, ...}

These are -NOT- examples:
{v5, v56, v573, v5732, ...}
(The first two digits of v573 are v57, not v56)
{v5, v56, v5674, v56746, ...}
(The second element has 2 digits and the third element has
4 digits meaning there is no element with a width of [3])

Set N5 is obviously a diagonal set; thus the set of all diagonal sets is not the null set.

Note: Kangas describes the posting of a diagonal set to (0,1) as a set of “drifting points.”

ISD Theorem: Given any ISD of R(0,1), there exists a unique diagonal set in Set Nv and Set Rv, where each element is used once and only once, which can exclusively be used by a sequence to define this element of R(0,1).

Proof:

The majority of the proof of this is trivial (by contradiction) because of the complete permutation nature of R(0,1) and Set Nv (note the Nv Theorem) and Set Rv, within their defined parameters and the above theorems. Proving every ISD has a diagonal set is similar to several other proofs in this paper and will be left to the reader.

The problem comes with the word "unique." No element of R(0,1) contains a unique FIS, meaning an FIS which no other element of R(0,1) contains. Nevertheless, in standard mathematics each ISD is considered unique. Thus given two unique ISDs, the two sets of FISs that make up each of these ISDs must also be unique, by definition.

Given any two ISDs, there must exist some first n, which is a DPI of both elements, for which the nth digit of the first ISD and the nth digit of the second ISD are not equal.

Thus they will have a different set of FIS's beginning with the nth FIS of each element. Note the set of FIS's of any ISD is in bijection with a subset of Set Nv or Set Rv, since these three sets are virtually identical. Therefore, the uniqueness of an ISD must imply the uniqueness of the set of FIS's that make up an ISD and this must imply the uniqueness of the diagonal subset of Set Nv or Set Rv which can also create this element.

In summary, if the infinite number of digits in an ISD is unique, then the infinite set of FIS's must be unique; thus the infinite subset of Set Nv must also be unique. **QED**

Thus we can take any element of R(0,1) and define a sequence which uses a unique infinite subset of Set Nv or Set Rv, namely a diagonal set, to build this element.

There are three reasons this is possible:
1) a diagonal set is an infinite subset of Set Nv or Set Rv, and
2) every element of a diagonal set has a different width, thus the nth digit of an ISD can be defined to equal the nth digit of the nth element of the diagonal set, and
3) both Set Nv, Set Rv and R(0,1) contain every possible permutation of digits within their defined bounds, meaning they have the same width and all have a Complete Permutation Construction.

Corollary to the ISD Theorem: Given any infinite and unique diagonal set, it will create a unique ISD.

Proof:

Given any two unique diagonal sets, d1 and d2, since they are different there must exist, by definition, an element of N, say n, such that the nth FIS of each set is different. The nth FISs of these two sets have a width [n] and have a different value. Since every element of N is a unique DPI of every ISD, this means that no ISD can be created by both d1 and d2 because its nth digit can only have one value. **QED**

Summary of Chapter XII:

Diagonal sets and the ISD Theorem are critical to the Creation Proof later in this paper. The ISD Theorem will be used to generalize the creation of 5/9 by Set N5 to apply to all of R(0,1). It is heavily related to the transfer of a set property (its width) to be the property of an individual element.

Chapter XIII: The Creation Definition:

The Creation Definition:

The Creation Definition basically states that if a set can be created by a countable number of operations, which meet certain criteria, then the set must be finite or have the same cardinal number as N. Here is the formal definition:

The Creation Definition (CD): "Let us take a finite number, c, and an empty set, S, and a sequence and algorithm based on N (defined as above). Let the first algorithm step create no more than c elements or partial elements of S (note: both elements and partial elements of S will be counted towards c). Let each subsequent and sequential algorithm step increase the cumulative cardinality of elements or partial elements of S by no more than c additional elements (ditto). If these conditions are met, then S is finite or countable."

The CD depends on a bit of common sense: "how can the cardinality of a set be greater than the number of algorithm steps (i.e. operations), multiplied by a finite constant, to create it?" N/o times c is N/o. In other words, how can there exist elements that had nothing to do with the algorithm steps?

Technically, c could be finite or countable, because if c is countable, by the CUT the entire set must be countable, but I have restricted c to being fixed and finite for my purposes.

If the sequence that creates a set creates a single distinct element of the set for each step, whether a finite expansion or an infinite expansion, then the set is obviously countable because the creation of the set comes with a trivial bijection with N built in.

But the CD does not come without some complexity. This complexity comes when a CD sequence creates a set of infinite expansions or infinite sets that, for any given step, are not "complete." In other words the sequence and algorithm "pieces together" the elements of the final sets, but no specific step completes some or any of the final sets. An example will help explain this situation.

The Set U/Set Uc Example:

Let us consider Cantor's CUT. In this theorem he used a Diagonal Serpentine Mapping (DSM) to create a single set, let us call it Set U, which is the union of a countable number of countable sets. Let us call the countable collection of countable sets: Set Uc, where the "c" stands for "collection."

Let us change things a bit and instead use Set U to **create** Set Uc. We note that the creation of Set Uc follows the criteria of the CD because there are a countable number of elements in Set U and there is one algorithm step or operation for each element of Set U. Each operation either starts the creation of a new set in Set Uc or it adds an element to a set that was created in a prior step. Thus the cumulative cardinality of Set Uc increases by 0 or 1 for any given step, and in all cases it is a partial new element (i.e. set). The CD criteria are satisfied.

It is obvious that since we are reversing the DSM that Set Uc consists of a countable number of countable sets and the collection of sets is therefore countable.

However, suppose we didn't know this and further suppose that we consider that no set in Set Uc exists until every one of its elements is created in toto by the algorithm. The reason for considering this last criterion is because of the DT. Every ISD can be viewed as an infinite set of digits. If we created the digits of an ISD by using an infinite sequence, which element of N "completes" the ISD and makes it a "fixed" number. Obviously there is none. Thus this scenario must be dealt with in some manner.

Let us take the first set of Set Uc and call it Set Uc_1. Which element of N, taken from the algorithm that creates Set Uc from Set U, completes the creation of Set Uc_1, or any other set within Set Uc?

It is obvious there is no specific element of N, taken from the creation of Set Uc, which completes the construction of Set Uc_1, et. al. (otherwise Set Uc_1 would be finite). Thus if we restrict our definition of a set within Set Uc to only "completed" sets, then no set within Set Uc will ever be created by a specific step! How then can we determine the cardinality of Set Uc?

By the CD we don't need to know which element of N completes each set within Set Uc, by the CD Set Uc is countable.

Understanding the Concept of "Append and Replace":

To understand the application of the CD to R(0,1) we must first consider the creation of the NN in the DT. In the DT every time a digit is appended to the NN the "old" intermediate value of the NN is "appended and replaced" or "converted into" a "new" intermediate value of the NN. For example, let us suppose that after the first 9 steps of diagonalization the value of the NN is .565555656.

This finite string, considered as of the 9^{th} step of diagonalization, is clearly not one of the first 9 elements of the original listing because of the way diagonalization works. It may be later on in the listing, in fact it may be the 10th element in the listing, but we know it is not in the first 9 elements. After step 9 of diagonalization we don't care where this element is in the listing. After the 9th step .565555656 is thrown away.

For example, let's say in step 10 it is "appended and replaced" by, say .5655556565. This number is not one of the first 10 elements of the listing, meaning it has not been mapped to by one of the first 10 elements of N.

In step 10 of diagonalization the first 9 digits of the NN are "appended and replaced" to become the first 10 digits of the NN. Without the appending and replacement operation the DT could not construct a single ISD. Rather it would create an infinite set of FISs.

Introduction to the Creation Algorithm:

The "Creation Algorithm" is an algorithm that creates all of the elements of R(0,1). It does this by an algorithm that is basically an expansion of the algorithm that creates 5/9 from Set N5. Instead of creating just 5/9, however, this algorithm will create all of the elements of R(0,1) simultaneously. It will do this with a countable number of operations in complete harmony with the Creation Definition.

In order to simplify and formalize the algorithm, it is necessary to introduce three terms, the last of which is the heart and sole of the Creation Algorithm. This last term is very complex at first glance, however, once it is seen graphically (as will be shown after the formal algorithm is defined) it is actually quite simple to understand. The reader might want to glance ahead several pages and look at the graphs before trying to tackle the third term (because of space the graphs are in base 3 instead of base 10).

The three terms are: Decaset, Decastring, and Multiple Digit Appending Operation Algorithm (MDAO).

Definition of Decaset:

Definition: A “decaset” (Base 10 Set) is a subset of Set Nv which meets these requirements:

1) Every element of the decaset has the same width, [w], and
2) If the width of the decaset is [2] or above, the first w-1 digits of each element of the decaset, called the "base" of the decaset, are equal, and
3) If the width of the decaset is [1], the "first w-1 digits," or "base" of the decaset, will equal a 'v'.

Obviously, every decaset has 10 elements. By ordering Set Nv in the same order as N, the first 5 decasets of Set Nv are:

1	v0, v1, v2, ..., v9	(base: v)
2	v00, v01, v02, ..., v09	(base: v0)
3	v10, v11, v12, ..., v19	(base: v1)
4	v20, v21, v22, ..., v29	(base: v2)
5	v30, v31, v32, ..., v39	(base: v3)

Definition of Decastring:

Definition: Whereas a "decaset" is a subset of Set Nv, a "decastring" is a single element of Set Rv which is equal to the "base" of the decaset, where the initial 'v' is converted to a decimal point.

To put this another way: given any decaset of width [w], the "decastring" is a string of digits which begin with a decimal point and has w-1 digits, such that the w-1 digits are equal to the base of the decaset without the 'v'. If the width of the decaset is [1], the "decastring" will equal a decimal point.

For example, consider the following decaset:

decaset = v97140, v97141, v97142, v97143, ..., v97149

The base of this decaset is = v9714 and the decastring of this decaset is = .9714

Decastrings will be considered to be elements of Set Rv, not Set Rt, in order to avoid problems with terminating zeros.

Definition of Multiple Digit Appending Operation Algorithm (MDAO):

Definition: A Multiple Digit Appending Operation (MDAO) involves two sets, a "first set" and a "second set." The "first set," for any specific step of the algorithm (but not necessarily the limit), always consists of elements of Set Rv or a single decimal point. The elements that make up the “first set” are constantly changing during the algorithm.

The "second set" consists of complete decasets that are elements of Set Nv. The “second set” does not change during the algorithm.

The MDAO is an algorithm that is defined to process every decaset in the "second set." This is how the MDAO algorithm works (don’t forget to look ahead at the graphs):

1) Given a "first set" which consists of elements of Set Rv or it consists of a decimal point, and

2) Given a "second set" which consists of complete decasets of Set Nv, ordered in the native order of N (note: the "second set" never changes). This is the order that these decasets are processed.

3) Consider a decaset of width [w], which is a subset of the "second set,"
Example: { ..., v98470, v98471, v98472, ..., v98479, ...} in "second set"

4) Find the element of the "first set" which is the decastring of this decaset,
Example: { ..., .9847, ...} in "first set"

5) Replace this decastring in the set with 10 decastrings equal to it, all of width [w-1]; meaning the set now includes 10 decastrings of this decaset instead of only one decastring,
Example: { ..., .9847[0], .9847[1], .9847[2], ..., .9847[9], ...}
Note: The number in the square brackets is an index and is not part of the number.
Note: Only the one decastring in the "first set" is affected by this operation.

6) For each consecutive element of the decaset, append its wth digit to the wth position of the wth consecutive decastring in step #5, giving this element of Set Rv a width of [w] and this element appends and replaces the decastring that was just appended,

Example:
1) (The first copy of the decastring) .9714[0] will have the 5th digit of v97140, a zero, appended to its 5th digit position, yielding .97140, which is now an element of the "first set" instead of .9714[0].
2) (The second copy of the decastring) .9714[1] will have the 5th digit of v97141, a one, appended to its 5th digit position, yielding .97141, which is now an element of the "first set" instead of .9714[1].
and so on.
Summary of Example: { ..., .98470, .98471, .98472, ..., .98479, ...} are now elements of the “second set.”

The phrase "appended and replaced" is equivalent to "converted into" or "transformed into." The original single decastring ceases to exist as an element of the first set after the MDAO because it is duplicated, appended and replaced, converted, and transformed into 10 strings of greater width.

In summary: { ..., .9847, ... } is "converted into":
{ ..., .98470, .98471, .98472, ..., .98479, ...}

Summary of Chapter XIII:

The Set U/Set Uc example demonstrates what can happen when an infinite set (or an infinite set of digits) is not considered to “exist” until all of its elements (or digits) are in place. The Creation Definition is a tool designed to deal with such paradoxical situations.

Chapter XIV: The Creation Algorithm:

The Creation Algorithm:

The Creation Algorithm is a sequence that defines or creates R(0,1). It will be a generalization of the sequence that uses Set N5 to create 5/9. To allow new notation, R(0,1) will simply be referred to as R. R will be defined by a sequence on Set Nv, **not** a sequence on N.

Before the sequence begins Set Nv will be ordered by size, as if they were elements of N (i.e. native order). The “second set” will consist of all of the elements of Set Nv in their native order. These elements will be grouped into consecutive decasets, meaning into consecutive groups of 10 consecutive elements each. Set Nv is the "second set" used by the MDAO and it will not change during the algorithm. The "first set" used by the MDAO will begin by containing a single decimal point: { **.** }. The contents of the "first set" will change for every decaset and MDAO.

Now the sequence will be defined:

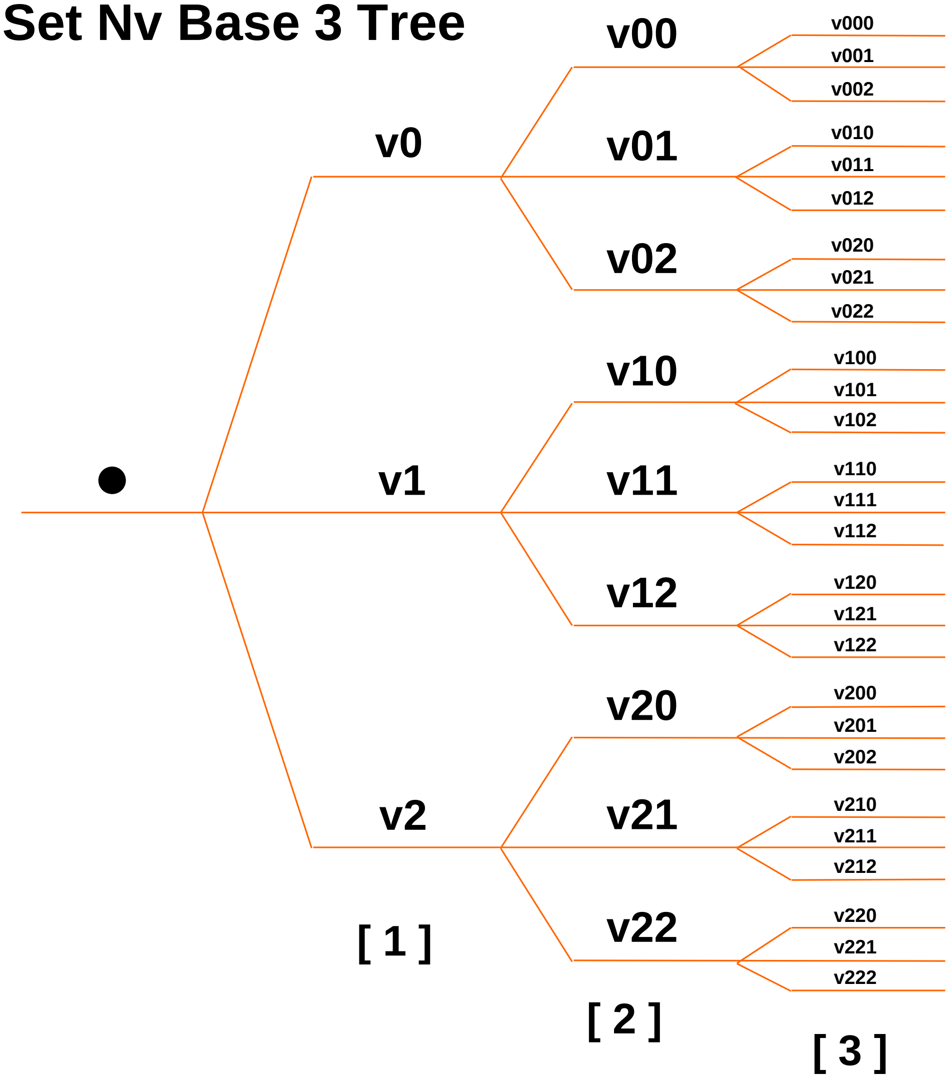
The Creation Algorithm: {all x | for every consecutive element of Set Nv, the “second set,” each consecutive decaset will be identified and the decastring of this decaset will be identified as an element of the "first set" (which begins as a single decimal point) and an MDAO will be executed}

The set defined by this sequence will be proven to be R(0,1) below. However, before these proofs, a visual display of what this sequence does will be shown.

The visual display will consist of a base 10 tree for Set Nv and the set created by the Creation Algorithm (CA). Because of space this tree will actually be viewed in base 3, which is actually the base used to initially visualize the CA.

This first graph is simply the "Set Nv Base 10 Tree." For the sake of space, this base 10 tree is actually shown in base 3. Note that the elements of Set Nv with a width of [1] are in the first "level," the elements of Set Nv with a width of [2] are in the second "level," and so on. By the Nv Theorem there is a level for every element of N.

Set Nv Base 3 Tree



Set R[]= { . }   (a decimal point)

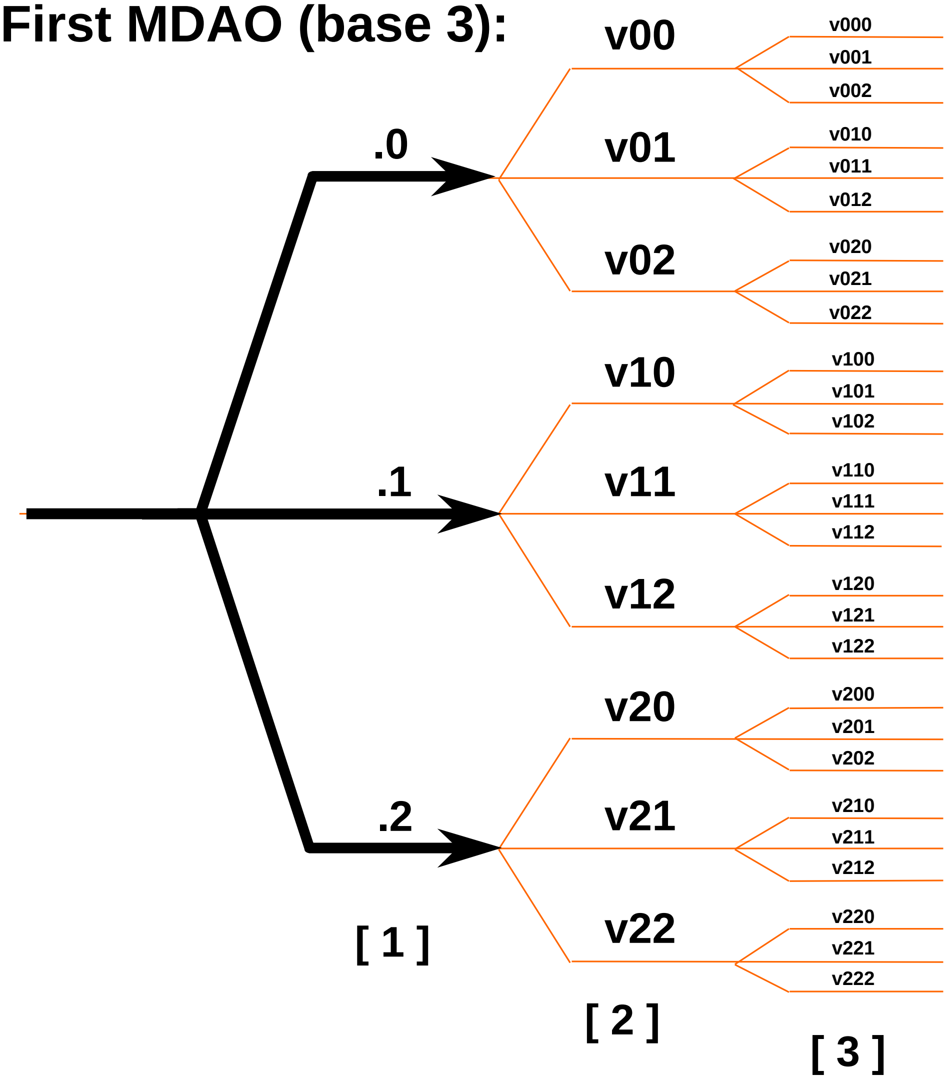
To expand on the terminology, each element of Set Nv sits on a "branch." Thus, in the third level there are 10^3 branches of the tree (base 10).

The first MDAO will use the decaset in the sequence that has a width of [1], thus the decastring is simply a decimal point, which not coincidentally is what the initial first set begins with: a single decimal point. The first decaset consists of v0 to v9, meaning an MDAO is executed on this decaset and R[] now equals:
{.0, .1, .2, .3, ..., .9}

The symbol "R[]" means: this is a snapshot of R at this point of the sequence.

Note that the "first set" started out originally as a single decimal point, but that the decimal point is no longer an element of the "first set." This is because the decimal point was converted into 10 decimal points and each of the 10 decimal points was "appended and replaced" by executing the MDAO.

First MDAO (base 3):



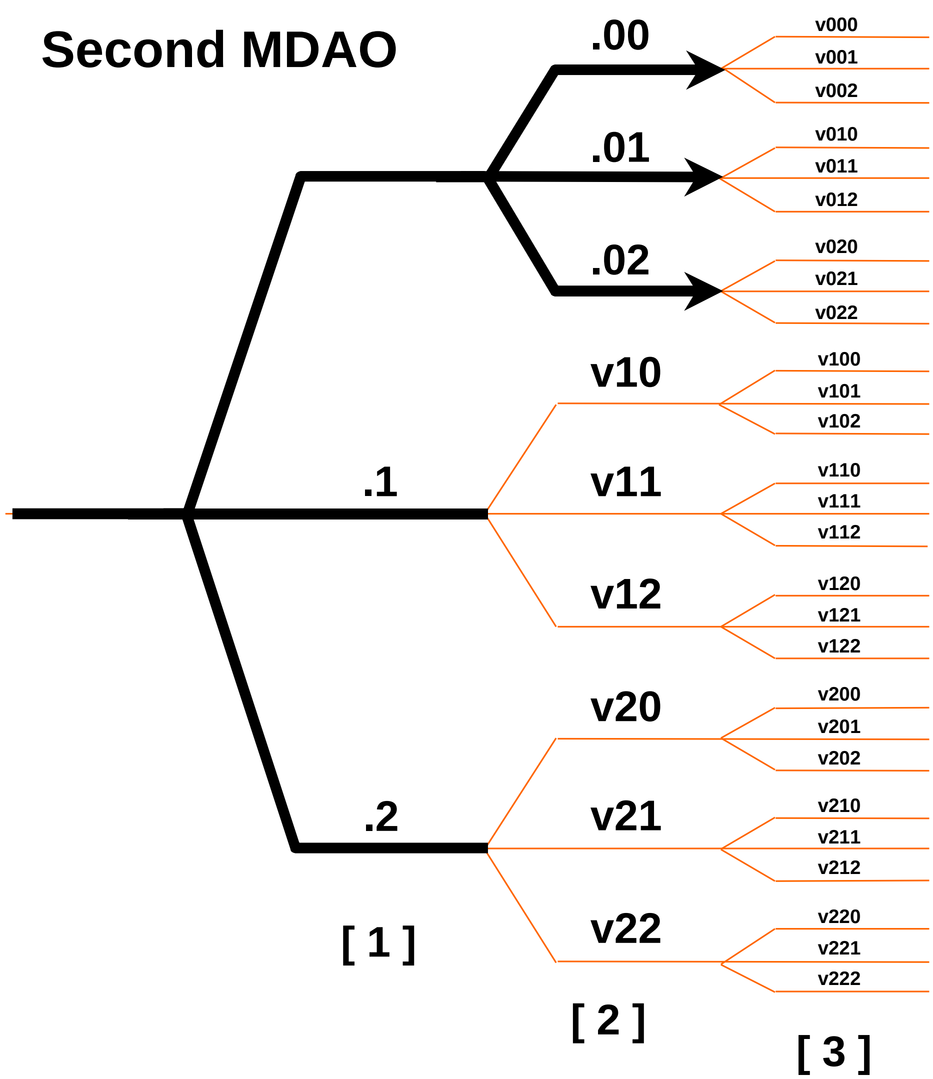
Set R[] = {.0, .1, .2}

The second decaset in the sequence are the elements: v00 to v09. After v00 to v09 have been executed by an MDAO, R[] equals:

{.1, .2, .3, ..., .9, .00, .01, .02, ..., .09}

Note that .0 was converted into 10 copies of .0, and each copy was "appended and replaced" by the MDAO, yielding the last 10 elements of R[] shown above. Note that .0 is no longer an element of R[].

Second MDAO

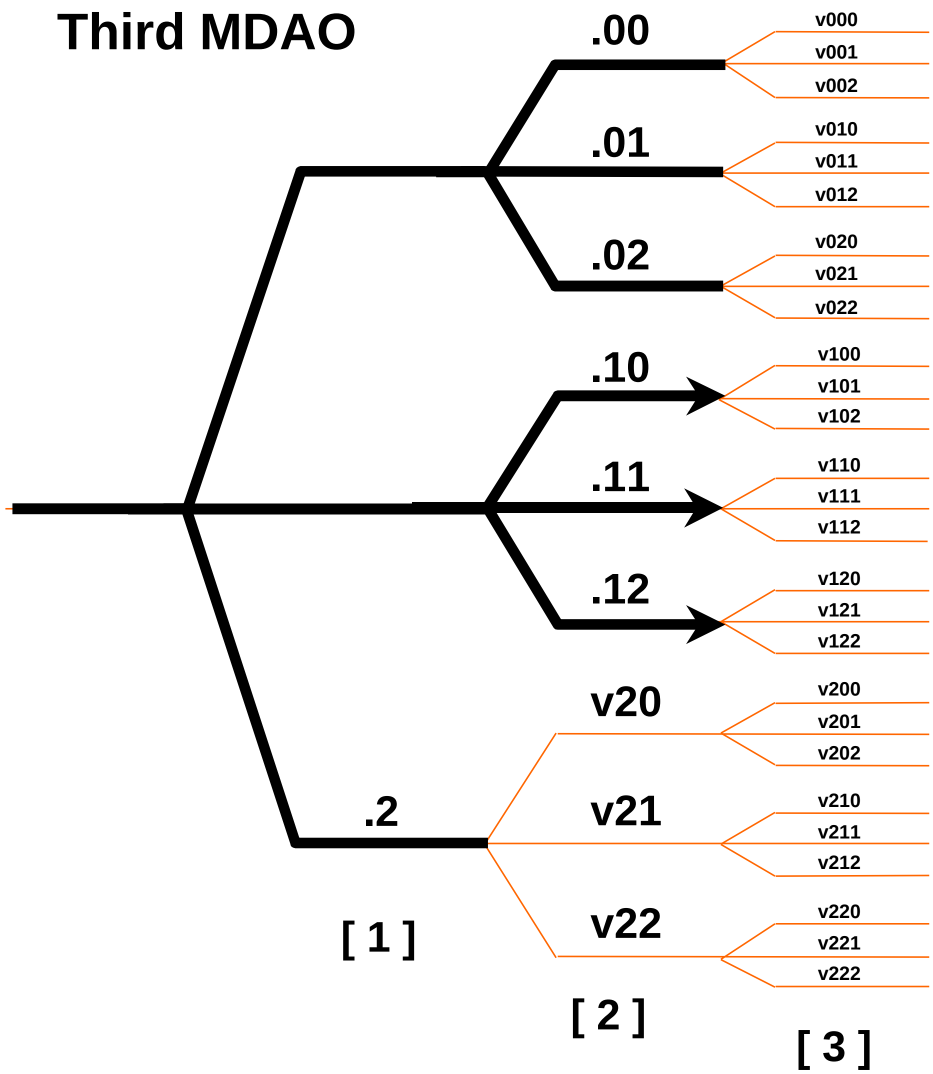


Set R[] = {.1, .2, .00, .01, .02}

The third decaset in the sequence are the elements v10 to v19. After v10 to v19 have been executed by an MDAO, R[] equals:

{.2, .3, .4, ..., .9, .00, .01, ..., .09, .10, .11, ..., .19}

Note that .1 was converted into 10 copies of .1, and each copy was "appended and replaced" by the MDAO, yielding the last 10 elements of R[] shown above. Note that .1 is no longer an element of R[].



Set R[] = {.2, .00, .01, .02, .10, .11, .12}

and so on for every element of Set Nv, meaning for every decaset of Set Nv.

Note that after the elements of Set Nv of width [1] have been executed by the CA, the cardinality of R[] is 10^1. After all of the elements of Set Nv of width [2] have been used by the CA, the cardinality of R[] is 10^2. And so on.

Summary of Chapter XIV:

The Creation Algorithm itself is only four lines long. But it took many pages to develop the concepts preliminary to it, going all the way back to the creation of 5/9 by Set N5. Likewise, it took several pages to graphically explain what it does. What it does, in one large “gulp,” is convert the width of Set Nv to be the width of the individual elements of a new set. The set turns out to be R(0,1) as will now be demonstrated.

Chapter XV: The Creation Theorem:

The Creation Theorem:

Two very different proofs of this next theorem will be given. One key concept that the reader should understand at the outset is that the set resulting from the above sequence is **whatever** a countable number of MDAO operations, operating on Set Nv, can generate in a single sequence.

It is claimed to be R(0,1), but whatever it is, it is created exclusively by a countable number of appending operations! The CD clearly states that the resulting set is countable, but the CD does not determine what the set contains.

The Creation Theorem: The result of the Creation Algorithm sequence is R(0,1) and R(0,1) is countable.

Proof #1:

Subproof 1: The Creation Algorithm (CA) creates a set that is not the null set:

The entire purpose of the above sequence is to convert a set property into an element property. No single element of Set Nv has a width of N, however, Set Nv as a set does have a width of N. The above sequence and appending algorithm is a generalization of the sequence that defines 5/9 by using Set N5 or the creation of the NN by the DT.

1) Because Set N5 is a subset of Set Nv, and because every element of Set Nv is used in the CA, therefore every element of Set N5 is used in the CA.

2) Because every element of Set N5 has a different width, every element of Set N5 is in a different decaset and therefore in a different MDAO (note: by definition every element of a decaset has the same width).

3) This means that Set N5, which has been shown to be infinite, is used in an infinite number of different MDAOs, each working on elements of a different width and each MDAO increasing the width of the elements that result from the algorithm.

4) Therefore N maps onto the "result" of the single or unique subset of appending operations that use the digits contributed by the elements of Set N5.

5) Since every MDAO appends a single digit to a single string, Set N5, if isolated in total in the CA, creates a string with an infinite number of digits.

6) By using logic similar to the proof that Set N5 can create 5/9, it is clear that 5/9 is a unique and valid element of the CA (also see the Corollary to the ISD Theorem).

7) Thus the set created by the sequence is not the null set because it contains 5/9.

Subproof 2: The set defined by the above sequence is countable:

1) Note that the above sequence is a sequence based on Set Nv, which means the sequence consists of a countable number of algorithm steps. Essentially, each 10 elements of Set Nv generate ten sequence steps and 10 appending and replacing operations (i.e. one complete MDAO operation).

2) For each element of Set Nv, meaning for each step of the sequence, there is exactly 1 digit made available for an appending operation (i.e. for each decaset of Set Nv there are exactly 10 digits made available for 10 appending and replacing operations - the MDAO). This establishes a bijection between Set Nv and the "set" of digits made available to the appending operations and in turn a bijection between Set Nv and the "set" of appending operations which are executed in the totality of the above sequence. Thus, 10 sequence steps equals 10 appending and replacing operations. Thus the number of "operations" is countable.

3) Because the cumulative cardinality of the resulting set increases by 9 elements for every 10 appending operations, therefore all of the criteria of the CD have been met and the resulting set is countable. Note that even the first 10 appending operations increase the cumulative cardinality of the set by 9 elements because the set starts out with one element - a decimal point.

Subproof 3: Every element defined by the above sequence is an ISD:

1) It has already been shown above that the set created by the CA is not the null set because 5/9 is created by Set N5. This same logic could be applied to any ISD in R(0,1) (this will be discussed in Subproof 4). In this subproof it will be shown that no element created by the CA has only a finite number of digits appended to it, meaning there are no "incomplete" ISDs or finite expansions in the set created by the CA.

2) Suppose there exists an element which results from the CA sequence that has only a finite number of digits appended to it, say n digits, meaning it has a width of [n]. "Width" in this case includes every digit appended to this element during the above sequence, even consecutive terminating zeros.

3) By random example, suppose this number was: .623...901, where the number of digits represented by the three dots is n minus 6. The last digit of this element was appended by the MDAO operating on v623...900 through v623...909, using .623...90 as the decastring.

If .623...901 is an element that results from the sequence we can conclude that the ten elements: v623...9010, v623...9011, ..., v623...9019 are not all elements of Set Nv. This guarantees that v623...9019 is not an element of Set Nv and it therefore guarantees that 5623...9019 is not an element of N (the initial 'v' was converted back to a '5').

4) Because n is finite, this means 5623...9019 is a finite expansion and therefore N must be finite. This is a contradiction. In other words, the set resulting from the CA contains only "complete" ISDs.

Subproof 4: Every possible element of R(0,1) results from the above sequence:

1) By the ISD Theorem, for every element of R(0,1) a sequence can define this element by exclusively using a diagonal set, which is a proper subset of Set Nv. Because every element of Set Nv is used in the above sequence, therefore, given any ISD, then every element of its diagonal set is used in the above sequence.

2) Because both Set Nv and R(0,1) have complete permutation constructions within their defined parameters, therefore, given any element of R(0,1), we can isolate the totality of elements of the diagonal set which can define this unique element of R(0,1), as we did with N5, and trace the elements of the diagonal set within the sequence, and in doing this we can trace the construction of this element of R(0,1) within the sequence (see the Corollary to the ISD Theorem). QED

Proof #2: Christensen's Proof

(**Note:** This proof is a modification of (Noel) Christensen's Paradox. A discussion of Christensen's Paradox, which has never been published, is beyond the scope of this paper.)

The part of this proof where the resulting set is shown to be countable by the CD is shown in the first proof. This proof will only be concerned that all of R(0,1) is created by the countable number of steps in the algorithm.

Let “R[n]” be the symbol for the snapshot of the CA after all of the elements of Set Nv of width [n] and less have been used by MDAO operations.

1) The first 10 elements of Set Nv can create the following set using the above sequence (i.e. the CA) (note that the first 10 elements of Set Nv have a width of [1]): R[1] = {.0, .1, .2, ..., .9}. Thus, because this is a CPC of width [1] (and because R(0,1) has a CPC), if the cardinality of R(0,1) is greater than what Set Nv can create, the elements of R(0,1) must have at least a width of [2].

2) The first 110 elements of Set Nv can create the following set using the above sequence (note that the first 110 elements of Set Nv have a width of [2] or less): R[2] = {.00, .01, .02, ..., .99}. Thus, because this is a CPC of width [2], if the cardinality of

R(0,1) is greater than what Set Nv can create, the elements of R(0,1) must have at least a width of [3].

3) The first 1110 elements of Set Nv can create the following set using the above sequence (note that the first 1110 elements of Set Nv have a width of [3] or less): R[3] = {.000, .001, .002, ..., .999}. Thus, because this is a CPC of width [3], if the cardinality of R(0,1) is greater than what Set Nv can create, the elements of R(0,1) must have at least a width of [4].

Let us continue this sequence and algorithm for all of the elements of N, meaning the width set of Set Nv.

The only way that Set Nv could not create R(0,1), with a countable number of appending operations, would be if the width set of Set Nv "ran out of numbers" before the width set of R(0,1) "ran out of numbers" or in other words, if there was a "gap" between the width of Set Nv and the width of R(0,1).

If both the width set of Set Nv and the width set of R(0,1) met all of the criteria of the Axiom of the Counting Numbers then they would be in bijection and the proof would be complete. Thus the width set of Set Nv must fail one of the tests of the ACN. It has been shown above, by the Nv Theorem, that Set Nv meets all of the tests of the ACN.

Thus, Set Nv can create R(0,1) and the criteria of the CD of "countable" have been met. Thus, R(0,1) is countable. QED

Comments on Christensen's Proof:

It is possible to modify Christensen's Proof so that it does not use the Creation Definition. To do this it is necessary to define one more term.

In the above discussion of the "Set Nv Base 10 Tree," the terms: "level" and "branch" were defined. We will now define the term "path."

A "path" is a consecutive and contiguous line drawn from the initial "v" through one branch in each level. For example, the path to v8743 **must** go from the initial "v," through the v8 branch, through the v87 branch, through the v874 branch, and will terminate at the v8743 branch. A path must be a continuous line. For example, we cannot draw a line from v877 to v8743.

Likewise we can look at "infinite" paths. A diagonal set of elements of Set Nv can be used to create pi/10, for example. This path would be defined as the path from the initial "v," through v3, through v31, through v314, through v3141, through 31415, and so on. From the above discussions, it is clear that every ISD can be represented by a unique, infinite path on the tree.

It is obvious that a finite path can represent every branch and an infinite path can represent every ISD.

Now consider this logic:

Level 1: There are 10^1 branches in level 1. There are also 10^1 paths to the elements in level 1.

Level 2: There are 10^1 + 10^2 branches in level 1 and level 2. There are 10^2 paths to the elements in level 2.

Level 3: There are 10^1 + 10^2 + 10^3 branches in the first 3 levels. There are 10^3 paths to the elements in level 3.

And so on.

In general there are 10^1 + 10^2 + ... + 10^n branches in the first n levels, but there are only 10^n paths to the branches in the nth level. For every element of N, meaning for every level number, except the first, the number of branches exceeds the number of paths. As the elements of N, meaning the level numbers, get larger and larger, the disparity between the number of branches and the number of paths diverges in favor of Set Nv.

In order for R(0,1), as a set of ISDs and infinite paths, to have a higher cardinality than Set Nv (the set of branches and finite paths), the width of the infinite paths much be much wider than the width of the branches. In other words, there must exist a digit position of an ISD that is not a level of the Set Nv base 10 tree. But from the above discussions: that is impossible.

Summary of Chapter XV:

The proofs above that R(0,1) is countable revolve around the concept that if a set can be created with a countable number of operations, and the cumulative cardinality of the new set increases by no more than a fixed finite amount for each step, then the set must be countable. The Creation Algorithm converts the width of a set, Set Nv, into the width of individual elements, namely ISDs. The CA creates all of the elements of R(0,1).

Chapter XVI: The Into-Definition:

The Into-Definition:

The CA uses the CD, but there is another definition of "countable" which it could just as easily have used, called the Into-Definition:

The Into-Definition: "Let us take an empty set, S, and a sequence and algorithm based on N (defined as above). After the first algorithm step let the cardinality of elements and partial elements be finite, meaning these elements and partial elements map "into" N (note: this is a pure "into" mapping, meaning not all of the elements of N are mapped to). For each and every subsequent and sequential algorithm step let the cumulative cardinality of the elements and partial elements be finite (ditto). If these conditions are met, then the cardinal number of S cannot exceed the cardinal number of N."

This definition is similar in form to this statement: "Let {a_1, a_2, a_3, a_4, ...} be an infinite sequence of points on R, where: 1) a_(n-1) is less than a_(n), and 2) a_(n), for any n, is less than k, a constant element of R. Then the limit of the sequence cannot converge to a number greater than k."

In other words, this definition states: if every element (i.e. the cumulative cardinality of the set being created) of an infinite sequence is **less** than N/o (aleph nought), then the limit of this sequence cannot be **greater** than N/o.

Just as plotting the elements of Set R5 onto the real number line cannot have a limit which "jumps" over 5/9; if the cumulative cardinality of a new set, for every element of N, is finite, and thus maps "exclusively into" N (meaning never "onto" for any step), the limit of the cardinality of the new set cannot "jump" over N/o and therefore the limit of the cardinality of the set must be finite or countable.

No specific proof that R(0,1) is countable will be given using the Into-definition. This is because the CA could have used the CD or the "Into-definition" to prove that R(0,1) is countable. Note that for every element of N the cumulative cardinality of R(0,1) (i.e. the snapshot R[]) is finite, and thus can map exclusively into N.

Note that we could have used the "increase" in cardinality instead of the "cumulative" cardinality in the "Into-definition." If the increase in cardinality, for every element of N, is finite, therefore for any finite step number the cumulative cardinality of the set must be finite.

All of this treats the cardinality of R(0,1) as a limit, meaning the limit of its width, because the cardinality of N, the width of R(0,1), is a limit. It also takes into account that the limit of the width of Set Nv and Set Rv are both N.

Summary of Chapter XVI:

In the last few parts we saw three new definitions of "countable," the PD, the CD and the Into-Definition. Each of these definitions are just as logical as the DOC. Each of them lead to proofs that R(0,1) is countable.

Chapter XVII: The Set R1 Paradox, Short N and Long N

The Set R1 Paradox:

In this part it will be proven that "long N" cannot map onto R(0,1). While it might appear this is a proof that R(0,1) is uncountable, such a conclusion would require a proof that the DOU, by itself (i.e. applied to "long N"), is true. In fact a paradox I call the Set R1 Paradox will lead to a proof that a pure DOU and the countability of R(0,1) cannot both be true. Based on other proofs in this paper this is a proof that the pure DOU is false.

All of this is a result of the fact that side-by-side mappings (i.e. sequential or rotating step mappings) between N and R(0,1) fail because of the properties of R(0,1), not because of the cardinality of R(0,1). What this means is that even if R(0,1) were countable, any side-by-side mapping between "short N" and R(0,1) or "long N" and R(0,1) will still fail.

What the concept of unsynchronized hinged sets is to "short N," the concept of the Set R1 Paradox is to "long N."

Set R1:

Set R1 was designed by Rusty Johnson to study R(0,1) in base 1:

Set R1 = { x | x is an element of R(0,1) and every digit of x is a 1 }
Set R1 = { .1, .11, .111, .1111, ... including 1/9 }

This base 1 set is obviously countable and contains a countable number of finite expansions and a single infinite expansion. Let us use the finite expansions in Set R1 to demonstrate how to create 1/9. We will do this similar to the way the DT creates the NN, meaning the nth digit of the NN is created in the nth step of the algorithm. In this case, however, the nth digit of 1/9 will be created in the nth step:

```
(1)     (2)       (3)
 N      Set R1    Construction of 1/9

 1      .1        .'1'     ('1' means newly appended digit)
 2      .11       .1'1'
 3      .111      .11'1'
 4      .1111     .111'1'
 5      .11111    .1111'1'
 ...
```

For each element of N in column #1, say n, there exists a finite expansion of Set R1 in the nth row that has a width of [n]. This is shown in column #2. This holds true for every element of N as has been shown many times above.

Column #3 shows the creation of the sole infinite expansion in Set R1, namely 1/9. For each and every element of N, n, the nth digit of 1/9 is "created." This is identical to the way the DT creates the NN.

Now let us look at column #2 as an attempt to be a complete listing of Set R1. Is it a complete listing of Set R1 elements, or to be more specific, is 1/9 a specific element of Column #2? The obvious answer is that no specific element of N in column #1 maps to 1/9 in column #2. If so, Set R1 would be finite. Furthermore, note that no specific element of N in column #1 maps to 1/9 in column #3. 1/9 is either a specific element in column #2 and column #3 (in the same row) or it is not a specific row in either column #2 or column #3. The latter statement is correct.

Let us summarize what was just said:

1) No specific element of N, taken from column #1, will "complete" the construction of 1/9 in column #3, meaning no specific element of N, taken from column #1, will "map" to 1/9 in column #3, and similarly
2) No specific element of N, taken from column #1, will map to 1/9 in column #2.

The phrase "taken from column #1" is absolutely critical to understand. When N is used to **create** the digits of an ISD (column #1 and #3), 1/9 in this case, no element of **this** N will map to that same ISD in the same listing (column #1 and column #2 or column #3).

Set R1 Paradox: "When the elements of N are used to create the digits of an ISD, no element of **this** N will map to the completed ISD."

The applications of the simple Set R1 Paradox are far and wide.

Comments on the Set R1 Paradox and "Short N"

In the DT, diagonalization synchronizes the N that is creating the NN and the column numbers of the R(0,1) array. See items #3 and #6 in "**The Six Synchronized Uses of Each n in N**." Since no element of N, taken from the N that is creating the NN, maps to the NN (by the Set R1 Paradox), then clearly no column number will map to the NN because these sets are synchronized. This also means that no DPI of the NN will map to the NN.

However, the DT also synchronizes the column numbers and the rows of the array. See items #3 and #4 in "**The Six Synchronized Uses of Each n in N**." This means that if no column number maps to the NN, then no row number will map to the NN!

The DT assumes that if R(0,1) were countable, N and the elements of R(0,1) could be listed side-by-side (CDOU). Suppose N were countable and was listed side-by-side with N. Having such a listing does not prevent us from attempting to apply diagonalization to this listing after it is shown to us. Diagonalization can be applied to any infinite listing, thus it could be applied to a side-by-side bijection between N and a complete listing of R(0,1) elements.

The DT makes certain assumptions as to what would happen if R(0,1) were countable and if diagonalization were then applied to this listing. It assumes:

1) If diagonalization were applied to this complete listing of R(0,1) elements; some specific element of N, say n, would be the row number of the completed NN created by diagonalization.

2) By synchronization, this would also mean that some column number of the array (i.e. some DPI of the NN) would also map to the completed NN.

3) By synchronization it would also mean that some element of the N that is creating the NN (item #6) would map to the completed NN.

By synchronization these are three different ways to state the same assumption. This assumption is not consistent with the Set R1 Paradox, which is true.

In other words, if R(0,1) were countable, diagonalization would work just fine by the Set R1 Paradox.

To prove that R(0,1) is countable, according to the logic of the DT, it would first be necessary to disprove the Set R1 Paradox. But the Set R1 Paradox is true, independent of the cardinality of R(0,1)!

The failure of the mapping between N and R(0,1) is caused by the properties of R(0,1) and the Set R1 Paradox, independent of the cardinality of R(0,1).

In other words, if R(0,1) were countable, diagonalization would work because of the Set R1 Paradox and the way that diagonalization synchronizes N in so many different ways.

While I could expand this discussion of the Set R1 Paradox and its relation to "short N," I have actually introduced the Set R1 Paradox to deal with "long N." This will be made apparent below.

Current Theorems That Use "Long N":

With the Set R1 Paradox in place, it is now possible to deal with the question of "long N." Both Cantor and Baire have theorems that deal with "long N." Cantor's proof was written in 1873/1874 (see the detailed proof in Dauben's book, pages 50ff) and uses nested intervals to discover (i.e. create) an element of R(0,1) that is not mapped to by any specific element of N. Baire's Category Theorem also uses nested intervals to prove essentially the same thing, though in a different way and by using different subset groupings.

Both theorems ignore the fact that there are two different Ns being used. If we look at the listing each deals with, we note that as the nested intervals converge that essentially the digits of the NN are being created. To say this another way, as the intervals get closer and closer together, more and more of the digits of the NN are "fixed" or "locked" into place, never to change again for the rest of the algorithm. It is similar to the way a power series can sequentially fix the digits of an ISD.

Thus as one N is mapping to the nested intervals and the listing (i.e. "long N"), the other N, "short N" or the digits/DPIs of the NN, are being expanded.

The two different Ns are hinged sets; however, in their theorems "long N" and "short N" are not assumed to be synchronized. But the fact that these two Ns are hinged, along with the Set R1 Paradox (which applies to "short N" and "long N"), essentially is the reason these proofs would work equally well whether R(0,1) were countable or uncountable.

Whether (0,1), the interval on the real number line, has a countable or an uncountable number of points, the nested interval proofs would work the same. Whether a countable or an uncountable number of intervals were devised, no element of Nc (i.e. the column numbers) is going to map to the NN. This is because of the Set R1 Paradox. All of this will be better understood after the "Long N Theorem."

Comments About "Long N":

Before the Long N Theorem, it is necessary to clarify exactly what is meant by "long N." Let "long N" be noted as LN and "short N" be noted as SN.

According to the DT, and the pure use of the DOU without diagonalization, if R(0,1) were countable, then N could map to all of the elements of R(0,1). Clearly the DOU means LN, not SN, because the DOU makes no reference to the columns of the array. But the issues surrounding LN are far from simple.

First, let me define what I call the "finite N principle," which has informally been mentioned before. Given any element of N, say z, z is a finite number, by definition, and thus a set that has a cardinality of z-1 must be a finite set. z cannot be preceded by an infinite number of other elements of N. This would require z to have an infinite number of digits.

To help understand the relationship between the "finite N principle" (FNP) and SN, suppose Set Rt is in a GTE Order. If it is then N cannot map to all of the elements of Set Rt. This N, however, is SN, not LN, because the width of the array is what is consuming N.

Now consider any element in the listing not mapped to by an element of N, such as .173 in the above example of the GTE Ordering. .173 is preceded by an infinite number of other elements of Set Rt. This N, however, is also SN for the same reason. However, the FNP rules out the possibility that any element of N maps to .173 because it would be preceded by an infinite number of other elements of N. Thus, not even LN can map to Set Rt.

Now take Set Rj in its logical ordering. N is consumed by Set R5, thus by the FNP no element of N maps to an element of Set Rnt. But R5 is an embedded, synchronized, hinged set and uses SN and thus Set R5 is not supposed to have an affect on LN. But it does because of the FNP.

But even if we ignore Set R5 (which is SN), Set Rt (which is LN) is infinite, without regard to synchronization or width. Set Rt is unsynchronized, Set Rj is countable, and Set Rj is in a logical order, yet LN is consumed even before the first element of Set Rnt is mapped to because of the FNP! Again, not even LN can map to Set Rj in a logical order!

LN is supposed to work on sets in a logical order. But it doesn't. LN is supposed to be impervious to the order of a set and to synchronization. But it isn't. LN, according to the DOU, should be impervious to the FNP. But it's not.

My point is that separating LN from SN is not as simple as it sounds because of the FNP.

Getting back to R(0,1), even if LN were attempted to map to a complete listing of R(0,1), by the FNP SN (i.e. the subset of R(0,1) mapped to by the elements of N used by diagonalization) consumes N (i.e. the FNP applies) before the NN is mapped to.

Whenever SN is used, the FNP principle seems to stifle LN.

That clearly was not the intent of a "pure" DOU.

The CUT is a logical choice to save the DOU because it does not use a character array; it uses an element array. However, trying to prove a set is countable by first placing all of its elements into a countable number of disjoint subsets is hardly a step forward.

In fact, the DOU cannot be salvaged. Even if we ignore the columns of the R(0,1) array, and simply go down the rows, we can still create a NN. Both Cantor and Baire proved that.

I will prove it again, but using a different technique, one that keeps track of both Ns.

Summary of Chapter XVII:

The Set R1 Paradox is very simple yet unbelievably powerful. By itself it makes a strong case that the DT does not prove that R(0,1) is uncountable.

Separating “short N” from “long N” is not as simple as it sounds. It is very difficult to separate SN and LN because of the FNP.

Chapter XVIII: The Long N Theorem:

The Long N Theorem:

Long N Theorem: These two assumptions cannot both be true: 1) the DOU using "long N" and 2) R(0,1) is countable.

Proof:

The wording of the theorem is an abbreviated way of saying that these two assumptions cannot both be true:

1) It is assumed that every countable set can be placed into bijection with "long N" (note: I will use a form of the DOU that ignores any direct mapping between the column numbers of the array and the rows of the array; but this form of the DOU does **not** ignore the FNP, meaning every element of “long N” must be preceded by a finite number of other elements of “long N"), and
2) It is assumed that R(0,1) is countable.

Taken together, these two assumptions mean there is an assumed bijection between LN and R(0,1), such that every element of LN that maps to an element of R(0,1) is preceded by a finite number of other elements of LN. No attempt will be made to map SN and LN to each other, but SN will be used to create the digits of the NN. In other words, LN and SN are treated independently.

A contradiction will be reached at the end of this proof. Since there are two assumptions, this means that at least one of these assumptions is false. This is not a proof that R(0,1) is countable or that the DOU is false.

Let us consider only the elements of R(0,1) that do not contain any '0's or '9's as digits. In other words, let us remove every element from the listing that contains one or more '0s' or '9s'. We will then recalculate the row numbers as Nr, meaning "long N."

I will call this technique "Quasi-Base 8," because the elements of this set (note: its elements are **viewed** as being in base 10) do not have the same properties as a pure base 8 set of ISDs would have. For example, it does not have any finite expansions, because no element contains a '0,' and it does not contain any "equivalent elements" (e.g. .2499999... and .25 are equivalent elements), because no element contains a '9'. This makes R(0,1) a lot cleaner set to work with and simplifies the proof.

Everything related to R(0,1), including the NN, will be in quasi-base 8. Everything related to N, Nr, Nc and "long N" will be in base 10.

In the algorithm we will construct a NN that is not mapped to by any element of Nr. The NN will not contain a '0' or a '9,' thus it will be a valid element/permutation of the set.

Consider any listing of R(0,1) elements (quasi-base 8) side-by-side with N (i.e. Nr or "long N"). The steps of the algorithm will be synchronized with Nr, meaning the algorithm will be executed for all of the elements of Nr, by definition.

N will be used in two ways: first, Nr maps to the elements of R(0,1), and second, Nc maps to the digits or DPIs of the NN. Nr and Nc will not be synchronized.

Note that the steps of the proof are driven by Nr, not Nc. The **logic** will follow the digits of the NN, but in fact it is the elements of Nr that are driving the algorithm steps.

Element #1 of Nr: Every element of R(0,1) (quasi-base 8) must have as its first digit one of these 8 permutations: {.1, .2, .3, .4, .5, .6, .7, .8}. Beginning with the first element, let us go consecutively down the Nr based listing until we find elements of R(0,1) that begin with 7 of these 8 first digits.

For example, suppose the first 108 elements of the list all begin with either: .1, .3, .4, .5, .7 or a .8. This means that none of these 108 elements begins with a .2 or a .6. Now let us suppose that the 109th element begins with a .2 (our knowledge that the 109th element completed the search for the 7 permutations is how the number '108' was chosen).

Now we will begin the creation of the NN. The first digit of the NN will be a .6. Nc now equals 1. Nr equals 109 and none of the first 109 elements begin with .6.

Element #110 of Nr: Beginning with the 110th element of the listing (in this example), let us go down the listing until we find elements of R(0,1) that begin with 7 of these 8 first **two** digits: .61, .62, .63, .64, .65, .66, .67 or .68. For example, suppose the elements of the listing between 110 and 257 all begin with either: .62, .63, .64, .66, .67 or a .68 or that the first digit of the elements is not a .6. This means that none of the first 257 elements in the listing begins with a .61 or a .65 (note that none of the first 109 elements began with a .6). Now let us suppose that the 258th element begins with a .65 (ditto to how '257' was chosen).

Now we will append a digit to the NN. The first two digits of the NN will be .61. Nc now equals 2. Nr equals 258 and none of the first 258 elements begin with .61.

Element #259 of Nr: Beginning with the 259th element of the listing (in this example), let us go down the listing until we find elements of R(0,1) that begin with 7 of these 8 first **three** digits: .611, .612, .613, .614, .615, .616, .617 or .618. For example, suppose the elements of the listing between 259 and 703 all begin with either: .611, .612, .613, .614, .615 or a .616 or that the first two digits of the elements are not .61. This means that none of the first 703 elements in the listing begins with a .617 or a .618. Now let us suppose that the 704th element begins with a .618 (ditto to how '703' was chosen).

Now we will append a digit to the NN. The first three digits of the NN will be .617. Nc now equals 3. Nr equals 704 and none of the first 704 elements begin with .617.

And so on for all of the elements of **Nr**.

Issue #1: First, it is necessary to determine whether a finite or an infinite number of digits are created for the NN (i.e. to be a valid element of R(0,1) quasi-base 8 the NN must have an infinite number of digits).

It needs to be proven that the NN cannot have a finite number of its digits. We will prove this by assuming both assumptions are true. There will be two proofs of this issue.

Proof 1a: Let us consider Set Rtt (see the discussion of Set Rj), which is known to be countable, in quasi-base 8.

Assume the width of the NN is [n], where n is an element of Nc. Set Rtt is a proper subset of R(0,1). Since, by assumption, Nr and R(0,1) are in bijection, and because the algorithm is executed for every element of Nr by definition, then every element of R(0,1) is mapped to by an element of Nr in the algorithm. Thus every element of Set Rtt is mapped to by an element of Nr. Thus, the algorithm is executed for all of the 8^(n+1) elements of Set Rtt (quasi-base 8) that have a width of [n+1]. By the Nv Theorem, from among the elements of Set Rtt can be found every possible permutation of [n+1] digits. Thus the NN cannot have a width of [n].

This is a contradiction. Thus the NN must have an infinite number of digits.

Proof 1b: Suppose the NN had a width of [n], where n is an element of Nc. This means by assumption and definition that for some row, say z, the nth digit of the NN was defined, and that no further digit of the NN was defined after z.

Let us assume the NN is equal to .617...836, as a result of step z, where the number of digits represented by the three dots is n-6. None of the first z elements in the listing, by construction, begins with this string.

Consider these eight subsets of R(0,1), namely all of the elements of R(0,1) that begin with: {.617...8361, .617...8362, .617...8363, ..., .617...8368}. If there was at least one element of R(0,1) mapped to by an element of Nr larger than z. from each of these sets, then the width of z would be at least [n+1]. This would be a contradiction.

Thus all of the elements of R(0,1) that begin with at least one of these strings, for example .617...8362, must be before z. There are two contradictions to this. First, no element before z is supposed to contain .617...836 as an initial substring, but every one of these elements contains this initial substring. Second, there are an infinite number of these elements; thus they cannot all be before z, by the FNP, because z is a finite number (note that this set is not associated with diagonalization). Thus the NN must have an infinite number of digits.

These proofs mean that the NN is a complete ISD and that Nc is equal to N.

Issue #2: Second, it is necessary to determine whether any specific element of Nc maps to the NN.

Since Nc is infinite, meaning it is equal to N, by the Set R1 Paradox it is clear that no specific element of Nc maps to (i.e. completes) the NN.

Issue #3: Third, it must be determined whether any element of Nr can map to the NN.

It has already been shown that the algorithm creates an infinite number of digits for the NN (by using the assumptions and Issue #1). By the way the algorithm works, this means that at least 8 times aleph nought (which is countable) elements of R(0,1) have been eliminated as being equal to the NN during the creation of the NN. No element of Nr can be preceded by 8 times aleph nought elements of Nr by the FNP. Thus no specific element of Nr can map to the NN.

Issue #4: Finally, what if the NN is created before all of the elements of R(0,1) are eliminated from being the NN?

Is it possible that the digits of the NN are completely constructed before all of the elements of Nr are executed? The answer to this is 'yes'. For example, list Set Rtt, in any order, before the other elements of R(0,1). However, the first assumption of this theorem is that every element of R(0,1) is mapped to by a finite element of Nr. If any element of R(0,1) existed in the listing after the NN was completely constructed; because it takes an infinite number of elements of Nr to create the NN, then the element of Nr that maps to this element must be preceded by an infinite number of other elements of Nr (the ones needed to construct the NN). This contradicts the FNP. This theorem is a search for whether Nr and R(0,1) **<u>can</u>** be placed into side-by-side bijection.

Conclusion of the Issues:

We have seen that the NN has an infinite number of digits, and that no specific element of Nc maps to (i.e. completes) the NN by the Set R1 Paradox. Also, because no specific element of Nc maps to the NN, then it takes an infinite number of elements of Nr to create the NN. Thus, by the FNP no specific element of Nr (i.e. LN) maps to the

NN. This is a contradiction to the assumption that both assumptions are true. This means that at least one of the original assumptions is false.   **QED**

Since this theorem is not a proof that R(0,1) is countable, it cannot be a proof that the DOU is false. It is simply a proof that both assumptions cannot be true. However, the other proofs in this paper that R(0,1) is countable resolve the issue as to which of the two assumptions is false.

Could this be a proof that R(0,1) is uncountable, meaning a proof that the DOU is true? To answer that question we must first ask ourselves the question: "if R(0,1) were countable, would the technique in the Long N Theorem still work?"

We found with the DT that every phase of the DT would work perfectly fine if R(0,1) were countable. Is the same thing true here?

Issue #1: If R(0,1) were countable Set Rtt would still be a proper subset of R(0,1). Thus the NN would have an infinite number of digits.

Issue #2: If R(0,1) were countable the Set R1 Paradox would still be true. The Set R1 Paradox is independent of the cardinality of any set; thus no element of Nc would map to the NN.

Issue #3: If R(0,1) were countable, because the NN has an infinite number of digits, it would still take an infinite number of elements of Nr to create the NN, thus no element of Nr would map to the NN by the FNP.

Issue #4: If R(0,1) were countable the same comments as above could be made.

Thus it is obvious that if R(0,1) were countable, every phase of the Long N Theorem would be true. The Long N Theorem cannot be a proof that R(0,1) is uncountable.

One last issue needs to be discussed. It has been shown that N and R(0,1) cannot be placed into bijection (or at least it has been shown that given any mapping a NN can be created). Does this mean the Cumulative Gap exists? The answer to this question depends on how "Cumulative Gap" is defined. If the existence of the Cumulative Gap implies R(0,1) is uncountable, then the Cumulative Gap does not exist, even though N and R(0,1) cannot be placed into bijection. If the Cumulative Gap means there does not exist a bijection, then it exists. Either way R(0,1) is countable and the DOU is false.

Summary of Chapter XVIII:

The Long N Theorem has shown that N and R(0,1) cannot be placed into bijection. But with that proof it is clear that this fact is not a proof that R(0,1) is uncountable. In fact, it is the DOU that is false. While “long N” avoids the problems of synchronized hinged sets, it does not avoid the problem that the R(0,1) array has two infinite orientations. When one of these orientations is shown to be infinite, by the construction of the NN, the other one must also be infinite. The Set R1 Paradox is to the Long N Theorem what the concept of unsynchronized hinged sets and the HSSDOU is to the DT.

Chapter XIX: The Continuum

About the Continuum:

With all of the problems caused by the width of R(0,1) being equal to N, it might be thought to change the way the elements of R(0,1) are represented in order to overcome these problems.

For example, suppose we represent every element of R(0,1) as a "Single, Unique Symbol" (SUS), perhaps something that looks like a single Chinese symbol as mentioned earlier. In this way the width of R(0,1) would be [1] and the DOU should work.

In fact it might be thought to make the DOU an axiom and to define N and R(0,1), as a set of SUSs, to be in bijection. Doing this might appear to avoid the problems caused by "hinged sets" and the "Set R1 Paradox.

But consider this ordering of the elements of R(0,1), as SUSs:

1) the elements of Set R5 by size, then
2) the rest of the elements of R(0,1) in any order.

If the elements of R(0,1) are represented as SUSs, will this be a valid mapping? The answer is obviously 'no'. All of the elements of N will be consumed by the elements of Set R5, even if its elements are represented as SUSs. This is because Set R5 is an infinite set and the FNP still applies.

The "continuum" itself, no matter how its elements are represented, has properties that virtually prohibit simple side-by-side or "spinning wheel" mappings between N and its elements. The "continuum" is a quagmire of quicksand, with trap doors. Baire's Category Theorem is an excellent and simple example of the paradoxes involved in trying to sequentially map N onto R(0,1).

The CA, on the other hand, is a very systematic approach to creating R(0,1). It is a simultaneous, synchronized and symmetrical technique to create the elements of R(0,1) from Set Nv. If not for the Set R1 Paradox, it would be a mapping technique that could map Set Nv onto R(0,1).

Likewise, in the section on the PD the mapping of Set Rt to (0,1) is a simultaneous, synchronized and symmetrical mapping technique. But even with such mappings the problem recurs to find a specific element of N that completes any ISD that is being "kept away" from the spinning wheel. It is like creating an apparatus that attaches to the body of a horse and hangs a carrot in front of its face. It can never "catch" the carrot no matter how many steps it takes because the body of the horse and the carrot are hooked together.

As we saw with the PD Theorem the elements of Set Rt and Set Rnt can be thought of as alternating with each other on the real number line (i.e. the base 10 tree example). So what is it about Set Rnt that prohibits a mapping with N, but allows N and Set Rt to easily be mapped together?

The answer is that Set Rt is composed of finite expansions, thus even though the width of Set Rt is infinite, its finite expansions can easily be ordered by width and placed into bijection with N. Furthermore, the Set R1 Paradox is not a problem with Set Rt.

With Set Rnt things are very different, particularly with the irrational numbers. We cannot actually identify all of the digits of any irrational number. For example, what is the last character in pi/10? How many digits does it have? It is this characteristic that an irrational number has a never-ending number of digits (and no one can tell what its actual, exact value really is) that prohibits the irrational numbers in Set Rnt from being ordered or all of them being mapped to by N. As Kangas said, an irrational number drifts forever. While R(0,1), even in quasi-base 8, can be placed into a logical order, a NN can always be created because of the spinning wheel.

The Pipe Mapping:

The Pipe Mapping is actually a way to order R(0,1) in a logical order, much like Set Rj was ordered. In this ordering we will look at R(0,1) in quasi-base 2, meaning we will only use two different characters, and we will look at the elements as being in base 10. We will use the characters '1' and '2'.

First Group: Let us look at all possible permutations of 1 digit position, taken from a pool of two characters: {1,2}. These permutations are {1,2}. To each of these strings we will add an infinite string of '1's and an infinite string of '2's, making 4 elements:

```
 N   R(0,1)

 1)  .1'11111...   (apostrophe added)
 2)  .1'22222...
 3)  .2'11111...
 4)  .2'22222...
```

Second Group: Let us look at all possible permutations of 2 digit positions, taken from a pool of two characters. These permutations are {11, 12, 21, 22}. To each of these permutations we will add an infinite string of '1's and an infinite string of '2's, making 8 elements. However, four of these elements are already in the listing, so only 4 more elements need to be added:

```
 N    R(0,1)

 5) .11'22222...   (apostrophe added)
 6) .12'11111...
 7) .21'22222...
 8) .22'11111...
```

And so on for all of the column numbers of the array (i.e. 1 group per column number):

After the nth group is listed, the cumulative cardinality of R(0,1), quasi-base 2, is: (2 * (2^n)).

This technique will list all of the elements of R(0,1) in a logical order. Does N map onto all of the elements of R(0,1)? Because of the Set R1 Paradox (and the FNP could apply), the answer is 'no'. No specific element of N will map to any of the irrational numbers and to many of the rational numbers (e.g. 1212121212...). This is because no specific element of N "completes" or "maps to" any of the irrational numbers and many of the rational numbers.

To put this another way, because no DPI of an irrational number, for example, maps to the irrational number, by the Set R1 Paradox, then no specific group number will map to, or include, this element. Thus no specific row number will map to it (proof by contradiction).

The Pipe Mapping is very similar to the orderings of Set Rj. With Set Rj the main problem was the FNP. However, if we had used the row numbers of Set Rj as the DPIs of any of the elements of Set Rnt, and tried to create one of these elements simultaneous with the mapping; then the Set R1 Paradox would have come into play and no specific row number would have mapped to this element. In other words, if we had "created" the elements of Set Rj, similar to the Pipe Mapping, no specific element of N would map to any of the elements of Set Rnt.

Applying Diagonalization to the Pipe Mapping:

Now that we have R(0,1) in a logical order, what if we apply diagonalization to the listing? I will not take the time for a thorough discussion of such an attempt, but it should be obvious to the reader why the digits of the NN fail to map to all of the elements of R(0,1).

For example, if the 1st digit of the NN is a '2', we note that many elements "later" in the listing also have '2' as a first digit. If the first two digits of the NN are '21', we note that many elements "later" in the listing contain '21' as their first two digits. And so on.

Diagonalization becomes a "game of keep away" with identifiable elements that exist "later" in the listing. I call the NN an "Infinitely Morphing Object."

If a game were set up between two people, one creating the digits of the NN, and one identifying, for every element of N, elements "later" in the listing that begin with this same FIS of width [n], who would win the game?

Summary of Chapter XIX:

Representing the elements of R(0,1) in different ways is not going to save the DOU. The properties of the “continuum” are such that N and the continuum cannot be placed into bijection. The Set R1 Paradox will forever prevent such a bijection as long as there are irrational numbers. Because of the irrational numbers drifting, and the spinning wheel, a NN can always be created.

Chapter XX: A Proof the DT and Power Set Theorem (PS) are Identical

Converting the Data in Both Proofs to the Same Representation:

(**Note:** When I originally wrote this section I was not aware that this result had been proven before. I cannot and do not, therefore, take any credit for this section.)

Most textbooks discuss the DT and PS in different chapters, as if they were entirely different proofs that R(0,1) is uncountable. This section will prove that they are identical proofs and that the disproof of one of them is a disproof of the other.

The Diagonalization Theorem (DT):

Let us consider a countable subset of R(0,1) as a set of ISDs in base 2, but viewed from the standpoint of base 10 (to avoid equivalent elements):

```
N          Element of R(0,1)  (base 2)

1          .0010011000100100...
2          .1001011011100111...
3          .0100111100111010...
4          .1111010101011110...
5          .1100010010101001...
and so on ...
```

If we apply diagonalization to this listing we can easily create a "new number," or NN, which is different than every element in the listing. The obvious algorithm is that the nth digit of the NN will be the opposite (meaning the opposite of '0' or '1') of the nth digit of the nth element in the listing. In this case the NN will begin as .11101 ...

Now let us look at this same set as a two-dimensional array of characters:

```
N          Element of R(0,1)

 Column # 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 ...

1          .0 0 1 0 0 1 1 0 0 0  1  0  0  1  0  0 ...
2          .1 0 0 1 0 1 1 0 1 1  1  0  0  1  1  1 ...
3          .0 1 0 0 1 1 1 1 0 0  1  1  1  0  1  0 ...
4          .1 1 1 1 0 1 0 1 0 1  0  1  1  1  1  0 ...
5          .1 1 0 0 0 1 0 0 1 0  1  0  1  0  0  1 ...
and so on ...
```

If we apply diagonalization to this two-dimensional array we will have the same results.

The Power Set Theorem (PS):

Now let us consider any countable subset of the Power Set of N:

N	Power Set of N
1	{ 3, 6, 7, 11, 14, ... }
2	{ 1, 4, 6, 7, 9, 10, 11, 14, 15, 16, ... }
3	{ 2, 5, 6, 7, 8, 11, 12, 13, 15, ... }
4	{ 1, 2, 3, 4, 6, 8, 10, 12, 13, 14, 15, ... }
5	{ 1, 2, 6, 9, 11, 13, 16, ... }
and so on ...	

Now let us look at the first set or element of the Power Set of N above, namely {3, 6, 7, 11, 14, ...}, in a different way. Rather than represent this set as shown, let us represent this set as a one-dimensional array.

Consider an array that has one row, but has a countable number of columns or cells, one column or cell for each element of N. Each cell of this array will have either a '0' or a '1' such that: if the nth element of N is an element of the set then the nth column or cell of the array will have a '1,' otherwise the nth cell will have a '0'. Since there is one cell for each element of N we can clearly represent any set of the power set of N as a one-dimensional array. For example, the first set above, namely {3, 6, 7, 11, 14, ...}, can be represented as a one-dimensional array as follows:

Cell #	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	...
	0	0	1	0	0	1	1	0	0	0	1	0	0	1	0	0	...

Now let us look at all of the elements in the above countable listing of the Power Set of N in a similar way. Each row will represent a one-dimensional array representing a set in the Power Set of N. The nth column will represent the nth cell of each one-dimensional array. Consider:

N	Power Set of N																
Column #	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	...
1	0	0	1	0	0	1	1	0	0	0	1	0	0	1	0	0	...
2	1	0	0	1	0	1	1	0	1	1	1	0	0	1	1	1	...
3	0	1	0	0	1	1	1	1	0	0	1	1	1	0	1	0	...
4	1	1	1	1	0	1	0	1	0	1	0	1	1	1	1	0	...
5	1	1	0	0	0	1	0	0	1	0	1	0	1	0	0	1	...
and so on...																	

Now let us apply the diagonalization algorithm from the PS. If the nth set or row in the listing contains n as an element (i.e. if the set in the nth row has a '1' in the nth column) we will fill the nth cell of the "new set," or NS, with a '0'. If the nth set or row in the listing does not contain n as an element (i.e. if the set in the nth row has a '0' in the nth column) we will fill the nth cell of the NS with a '1'. The NS thus begins: 1 1 1 0 1 ...

By now the similarities between R(0,1) in base 2, with every element viewed as a base 10 ISD, and the Power Set of N, where each set is viewed as a one-dimensional array, should be obvious.

Every permutation of R(0,1) in base 2, looked at in base 10, can clearly be represented as a set within the Power Set of N. Likewise every set within the Power Set of N can be represented as an ISD of R(0,1). The concept of "permutation" and the concept of "power set" are identical using this common representation.

Clearly, disproving the DT is the same thing as disproving the PS.

Summary of Chapter XX:

Cantor's Diagonalization Theorem and his Power Set Theorem can each have their respective data translated into a common method of representation. While these two theorems look at sets in a very different way, they are in fact equivalent theorems. Thus the disproof of one of them is a disproof of the other.

Chapter XXI: Concluding Comments

This paper has done everything necessary to prove that R is countable and thus solve the Continuum Hypothesis. I have shown that the underlying assumption of the CH (that N and R have different cardinal numbers) is false.

This paper has isolated the error in the DT, it has explained why the DT is false, and it has proven that the DT would work the same whether R(0,1) were countable or uncountable. It has developed three very logical and reasonable definitions of "countable" and has used these definitions to prove that R is countable. It has also disproven the DOU and dealt with "long N."

Many papers use the DOU/diagonalization combination directly or indirectly, and/or ignore the Set R1 Paradox or simply assume that R(0,1) is uncountable before they begin. Theorems by Cantor, Goedel, Cohen, Turing, Baire, Church, Borel and several others include some relationship with uncountability. All of their theorems need to be rewritten based on the results of this paper.

(Note: As I understand it Turing did not claim there were an uncountable number of Turing Machine programs, he simply referred the matter to Church's Thesis.)

The conclusions of this paper do not mean that these other papers do not contain many valuable concepts, they each contain valuable concepts; this paper simply proves that they do not prove that certain sets or systems are "uncountable" or "incomplete" or "not computable."

End of Paper

Credits:

Many thanks to (listed chronologically in order of their assistance and what they were doing when their contributions were made):

1) Rusty Johnson (MS from Kansas State University, Mystech renamed Sterling Software)
2) George Kangas (ABD Grad Student, University of Kansas)
3) Ed Schmeichel (Professor, San Jose State University)
4) Jack Lamoreaux (Professor, Brigham Young University)
5) Will Schneeberger (Grad Student, Princeton University)
6) Noel Christensen (MS from University of Utah, Allied Signal)

Not all of the contributions made by these individuals are mentioned in this paper, nor are all of those who made contributions listed. Many contributions to this paper were steps in a process and these processes were eventually superseded by the things that did end up in the paper.

Nevertheless, each of these individuals made significant contributions to this paper for which I am grateful.

As always, using a quote from another mathematician or noting appreciation for their assistance does not constitute an endorsement by that mathematician or their institution.

About the Author:

Robert Webster Kehr was born in Jefferson City, Missouri (USA) in 1946. He has studied mathematics at three universities and has studied Set Theory since 1964. Since 1972 he has specialized in Transfinite Set Theory and Transfinite Permutation Based Number Systems.

Webster is the author of Kehr's Paradox, a TST theorem not used in this paper, and Kehr's Observation, a problem in combinatorics that was solved by his long time friend Ed Schmeichel of San Jose State University.

Webster and his wife Marit have been married for 27 years, have 7 children and 6 grandchildren. They live in Overland Park, Kansas (USA) with 4 of their children. Webster is an ex-Marine, Viet Nam veteran, sings in a regional church choir and works at Sprint Corporation in Kansas City, Missouri (USA).