Link Theory -

From Logic to Quantum Physics

Richard Shoup and Thomas Etter
Boundary Institute
Los Altos, California
rshoup@boundary.org, etter@boundary.org
revised August, 2008

Abstract: This informal paper presents the basic ideas and mechanics of Link Theory, along with several examples from diverse domains including logic and digital circuits, arithmetic and statistical reasoning, and quantum physics and quantum computing.

0. Introduction

0.1 What is Link Theory?

Link Theory is essentially a calculus of composite abstract relations. As such, it is *not* itself a theory of computation or physics (or anything else), but a *tool* for creating and analyzing structures which represent digital circuits, physical situations including quantum physics experiments, and much more. Several examples are given below. Link Theory was developed primarily by Thomas Etter beginning in 1996 (Etter & Noyes, 1998), but it has origins in much earlier work as well.

A *relation* is a set of mutually consistent *possibilities* among several variables. This set is in principle finite. Link Theory shows how to represent relations in tabular form, and how to compose relations into larger relations. A relation may also be viewed as a *constraint* on the set of all possible mutual values of several variables.

By *abstract* relation, we mean something which represents structural properties of a relation that don't depend on the kinds of things which are thus related. This means, for instance, that the structure of the Boolean relation (A = B AND C), as given by a table containing values 0 and 1, could just as well be given by a table with values "true" and "false", or "good" and "bad", or "Mars" and "Venus", or 0 and 1 reversed. The abstract AND relation is that which all such tables have in common.

1. Link Theory – the basics

In this section, we describe the mechanics of Link Theory and give several examples of its use in representing digital logic circuits.

1.1. Variables, domains, cases, case counts

To get started, we take the simple binary domain of digital logic. Each *variable* or signal can take on either of two *values*, which we designate as **0** and **1**. The value of a variable is seldom in any sense actual or definite in this calculus, but rather *the values represent possibilities*. It is from a simple enumeration of possibilities which we will build our universe.

Here is a set of common logic gates, each represented by a table consisting of *cases*, or rows, which enumerate all possible combinations of values among the variables indicated. Note that this representation is entirely symmetric, and no distinction is made between inputs and outputs. Unlike real electronic gates, no direction of action or causal relationship is implied or enforced here.

x	y	n
0	0	0
0	1	1
1	0	1
1	1	0

x	y	z	$n_\wedge$
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

x	y	z	$n_\vee$
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

Each table also includes an extra column labeled **n**, called the *case count* for that row. The case count may be thought of as the number of ways in which that combination of values can occur. In our initial examples, a case count will usually be 0 (impossible) or 1 (possible). A short inspection of the tables above will show that 1s in the case count columns correspond to the usual functions of these familiar gates. (It is important to distinguish the logical *values* designated here as **0** and **1** from the case counts 0 and 1, which are *numbers*. We could have used another nomenclature, but both uses are standard and familiar to the digital engineer.)

In general, case counts can be *greater than 1*, indicating hidden or "don't-care" variables, and *less than 0*, indicating quantum states and situations. The meaning of a *negative* *possibility* is indeed interesting to contemplate, but as a philosophical matter is mostly beyond the scope of this paper.

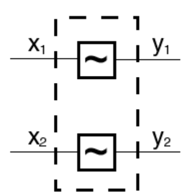
1.2. Relations

A Link Table may also be regarded as a *relation* or *constraint* on the joint values of its variables. Those cases which have a case count > 0 are permitted, that is *valid* under the constraint, those with case count = 0 are disallowed or *impossible*. A relation is inherently *symmetrical*, having no "input" or "output", unlike our usual conception of digital circuits. Relations are in general not functions in the mathematical sense, but all functions are relations. The bi- (multi-) directional character of relations can be used to great advantage in synthesizing and analyzing circuits.

1.3. Composing relations

Composing Link Tables is a simple matter of combinatorics -- accounting for all the possibilities open to the combined structure. A composite *linked* relation or constraint is a *limitation* on the set of all the *joint possibilities* for the combined set of variables.

For example, if we consider two NOT gates (inverters) taken together (but not yet connected), there are a total of four binary variables represented by their terminals or wires, and thus 16 total cases in the Cartesian product. But since both NOT constraints must be satisfied, only 4 cases are actually valid. *The case count in the composite table is derived by simply multiplying the case counts of the components for each case.*



x_1	y_1	n_1
0	0	0
0	1	1
1	0	1
1	1	0

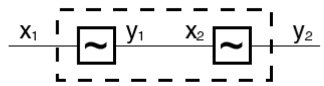
$\otimes$

x_2	y_2	n_2
0	0	0
0	1	1
1	0	1
1	1	0

=>

x_1	y_1	x_2	y_2	n
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	0
1	0	0	0	0
1	0	0	1	1
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

Suppose we now connect the two circuits serially. The diagram below again shows two NOTs, but with the addition of a connection, or *link*, between variables $\mathbf{y}_1$ and $\mathbf{x}_2$. In the composite table, these two variables must therefore agree in all cases, and 8 invalid cases have been crossed out. Note that only two of these were valid in the first place.



x_1	y_1	n_1
0	0	0
0	1	1
1	0	1
1	1	0

⊗

x_2	y_2	n_2
0	0	0
0	1	1
1	0	1
1	1	0

=>

x_1	y_2	n
0	0	0
0	1	0
~~0~~	~~0~~	~~0~~
~~0~~	~~1~~	~~0~~
~~0~~	~~0~~	~~0~~
~~0~~	~~1~~	~~1~~
0	0	1
0	1	0
1	0	0
1	1	1
~~1~~	~~0~~	~~1~~
~~1~~	~~1~~	~~0~~
~~1~~	~~0~~	~~0~~
~~1~~	~~1~~	~~0~~
1	0	0
1	1	0

=>

x_1	y_2	n
0	0	1
0	1	0
1	0	0
1	1	1

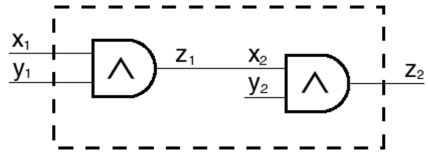
$$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \Rightarrow \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Furthermore, we don't care at the moment what the values of $\mathbf{y}_1$ and $\mathbf{x}_2$ are, so we *hide* these variables. That is, we do not connect them outside the new composite circuit (dashed box). Variables $\mathbf{x}_1$ and $\mathbf{y}_2$ remain, and *are* of interest and visible outside the box. To get the overall result, we can simply erase the column associated with the hidden variable (the composite $\mathbf{y}_1$- $\mathbf{x}_2$), and *sum the case counts of like cases*.

Four cases of the two variables remain, each with two instances in the composite table. Summing, only two cases are valid (n > 0). As expected, linking two NOTs in this trivial way yields the *identity relation*, since only the cases **0-0** and **1-1** have case counts = 1.

Note that we have included at the bottom of the figure an equivalent matrix multiplication, where the case counts are taken as matrix coefficients, each indexed by the corresponding logic values. Later we show the equivalence of linking to matrix (more generally, tensor) multiplication.

Variables may be hidden or disregarded, as above, for various reasons. In the example below, we want to link the output of the first gate ($\mathbf{z}_1$) to one of the inputs of the second gate ($\mathbf{x}_2$), and then determine the relation which results between inputs $\mathbf{x}_1$ and $\mathbf{y}_1$ and the output $\mathbf{z}_2$. We choose to hide or ignore both the linked variable $\mathbf{z}_1$- $\mathbf{x}_2$ and also the other input $\mathbf{y}_2$. (An engineer would call $\mathbf{y}_2$ a "don't care".) To do this, we link variables $\mathbf{z}_1$ and $\mathbf{x}_2$ in the AND gate tables as above, then hide (erase) the columns for $\mathbf{z}_1$- $\mathbf{x}_2$ and $\mathbf{y}_2$, and combine like cases by summing their case counts.



x_1	y_1	z_1	$n_\wedge$
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

$\oplus$

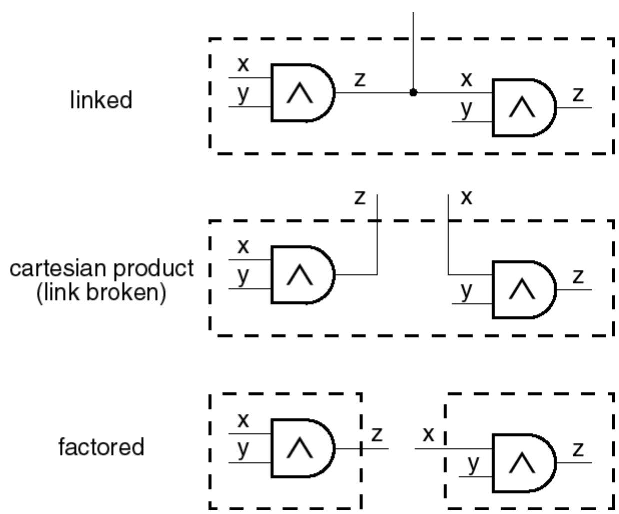
x_2	y_2	z_2	$n_\wedge$
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

=>

x_1	y_1	z_2	n
0	0	0	2
0	0	1	0
0	1	0	2
0	1	1	0
1	0	0	2
1	0	1	0
1	1	0	1
1	1	1	1

1.4. Breaking a Link

Links may also be *broken*, and components which are thus separated are called *factored*. Below we show two AND gates linked much as before, but this time with only the linked variable **z-x** visible. (We have hidden all the others just to simplify this example. The reader may verify that case counts of 6 and 2 are the result.) We then break the link **z-x**, thus separating the variables **z** and **x** again. What remains is the Cartesian product of the two tables, with only **z** and **x** visible. (In physics terms, this is the *density matrix* relating the two variables) Finally, we separate the two components into two tables whose product is the composite table. Of course, not all such tables (circuits) are factorable, as we will see below in the cases of quantum circuits.



zx	n
0	6
1	2

z	x	n
0	0	6
0	1	6
1	0	2
1	1	2

z	n
0	3
1	1

x	n
0	2
1	2

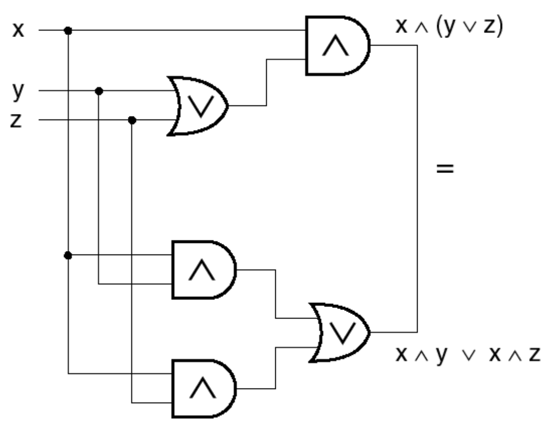
2. Examples and extensions

Here are some other examples including logic, arithmetic, and quantum circuits.

2.1 Example: Distributive Law

Below is somewhat larger example, again in the logic domain. By combining instances of the tables given above, and linking variables as previously described, we arrive at a simple "proof" of a Distributive Law of logic (AND distributes over OR). To represent the claimed equality, we simply connect (link) the two expressions together – something ordinary circuits (and most engineering design tools) would not allow.

For every combination of values for **x**, **y**, and **z**, the composite relation is valid, as shown by the resulting case counts (all 1s). If the circuit did not yield a tautology, the counts would indicate exactly those cases where the theorem failed.



x	y	z	n
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

2.2. Example: Arithmetic

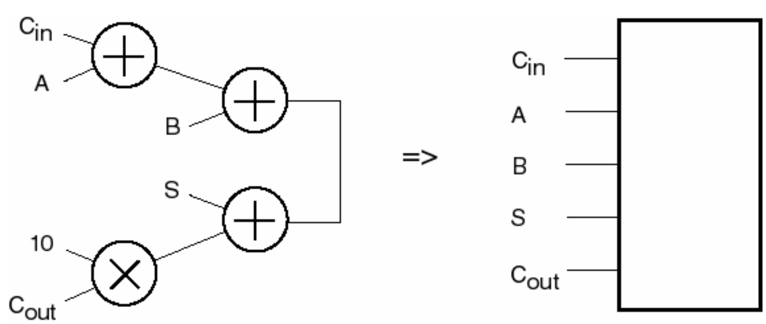
Link Theory has been successfully applied not only to logic domain problems, but also to numerical problems such as cryptarithms (shown here), finite-domain algebraic problems such as Diophantine (integer-solution) equations, and more. Here is a cryptarithm example usually considered in the domain of Artificial Intelligence. The problem is to find an assignment of single digits 0-9 to the letters such that the overall sum is satisfied. Each letter must correspond to a unique digit.

$$\begin{array}{r} \text{SEND} \\ +\text{MORE} \\ \hline \text{MONEY} \end{array} \quad \Rightarrow \quad \begin{array}{l} D + E = Y + C_1*10 \\ C_1 + N + R = E + C_2*10 \\ C_2 + E + O = N + C_3*10 \\ C_3 + S + M = O + C_4*10 \\ C_4 = M \end{array}$$

with $S,E,N,D,M,O,R,Y \in \{0\text{-}9\}$ uniquely, and $C_i \in \{0,1\}$

Shown are the simultaneous equations implied by this sum, one equation per column. $\mathbf{C_1}$ represents the carry out of the right-most column, and so on. We also assume that positional numbers do not begin with a leading 0, so **M** must equal 1.

To solve this problem using Link Theory, we first create a Link Diagram to represent each column constraint, as shown below. Each column is made up of three addition tables (two inputs, only up to a sum of 19) and one multiply-by-10 table. Compositing this Link Diagram results in a single 5-input Link Table for each column.



$C_{in} + A + B = S + C_{out}*10$
for each column

By linking together 4 such column tables according to the problem (linking **E** in column 1 to **E** in columns 2 and 3, for example), applying the uniqueness constraints (**S≠E**, **S≠N**, **S≠D**, … , **E≠N**, **E≠D**, … , **R≠Y**) with tables similarly, and reducing the complete composite Link Table, we are left with a table containing only one case, the solution:

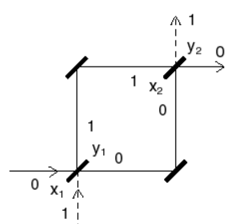
$$\begin{array}{r} \text{SEND} \\ +\text{MORE} \\ \hline \text{MONEY} \end{array} \quad \Rightarrow \quad \begin{array}{r} 9567 \\ +1085 \\ \hline 10652 \end{array}$$

2.3. Example: Quantum interference

In this example, we introduce a "gate" whose table includes a *negative case count*. (Another name for this might be a "possibility debt".) The diagram below represents a Mach-Zehnder interferometer consisting of four mirrors -- two half-silvered and two fully-silvered. Four variables will be used to describe the possible paths of a photon through the apparatus.

A photon enters from the left (signified by variable $\mathbf{x}_1 = 0$) or from below ($\mathbf{x}_1 = 1$) and impinges on the first half-silvered mirror. Variable $\mathbf{y}_1$ indicates the direction in which the photon leaves this mirror (upward, $\mathbf{y}_1 = 1$, or to the right, $\mathbf{y}_1 = 0$). The photon then bounces from a fully-silvered mirror and arrives at the second half-silvered mirror from the left ($\mathbf{x}_2 = 1$) or from below ($\mathbf{x}_2 = 0$), and exits upwards ($\mathbf{y}_2 = 1$) or to the right ($\mathbf{y}_2 = 0$). (For this example, we have simplified the actual physical situation and neglected complications introduced by the fully-reflecting mirrors.)

Using negative case counts, we can model a crucial aspect of the situation which actually occurs in nature. Tables for the two half-silvered mirrors are shown, along with their composite table, and the corresponding matrix representation. From the tables or the matrices, it is easy to see that the half-silvered mirror acts as a 45° *rotation* (phase shift) on the photon state vector. In quantum physical terms, we have applied a *unitary transform* to the photon, and created a *superposition* of the two paths through the apparatus.



$$\begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix} \Rightarrow \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$$

x_1	y_1	n_1
0	0	1
0	1	1
1	0	-1
1	1	1

$\oplus$

x_2	y_2	n_2
0	0	1
0	1	-1
1	0	1
1	1	1

=>

x_1	y_2	n
0	0	1
0	1	-1
0	0	1
0	1	1
1	0	-1
1	1	1
1	0	1
1	1	1

=>

x_1	y_2	n
0	0	2
0	1	0
1	0	0
1	1	2

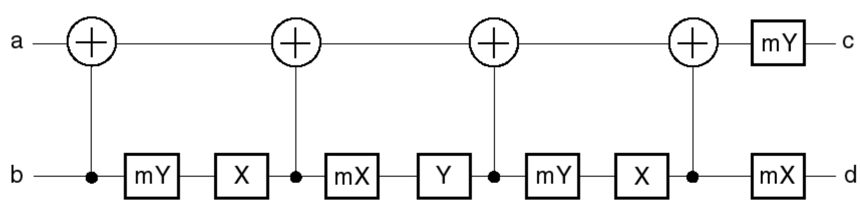
By hiding the linked variables $\mathbf{y}_1$ and $\mathbf{x}_2$, we are left with two cases for each of the four values of $\mathbf{x}_1$ and $\mathbf{y}_2$, similarly to the previous example. However, in two cases the negative values cancel with the positive values, and the result is 0, or impossibility -- in physical terms, *destructive interference*. In the other two cases, the counts add, i.e., *constructive interference*. Thus, from the composite table we can see that any photon entering from the left *must* exit from the right, and a photon entering from below will necessarily exit upwards.

2.4. Quantum computation

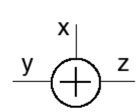
Numerous research groups have experimented with computational devices which operate in the quantum realm, in hopes of building a *quantum computer*. A variable in a quantum computer can in principle represent, and compute upon, a superposition of several values at once.

The following quantum computer circuit (or algorithm, equivalently) was suggested by Isaac Chuang of IBM Almaden Research Center. Using Link Tables with case counts as shown for each operator as shown below, we calculate the composite transformation as previously described, and the result is identical to that obtained using standard quantum matrices. Details of the result are given in Appendix B, and the industrious reader may verify this by compositing the given tables as shown in the diagram.[1]

Note that the imaginary case counts of $+i$ and $-i$ required by the given X and MX component operators can be accommodated by introducing two new variables linked in common among all elements, thus quadrupling the number of cases. This is equivalent to substituting a 2x2 spinor with real values for the imaginary matrix coefficients, see Appendix A. This shows that imaginary values are *not* needed in quantum mechanics (Mackey, 1963), nor as a further extension to Link Theory, and this fact may be a clue to deeper understanding of the underlying structure of this formalism.



x	y	n_Y	n_{mY}	n_X	n_{mX}
0	0	1	1	1	1
0	1	1	-1	i	$-i$
1	0	-1	1	i	$-i$
1	1	1	1	1	1



x	y	z	n_{CNOT}
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

[1] A complete Link Theory simulator/calculator has also been implemented, and has been used to verify all the examples herein.

3. The bridge to conventional physics notation

In a physicist's terms, the general binary pure state (in Dirac notation)

$$a|\mathbf{0}\rangle + b|\mathbf{1}\rangle$$

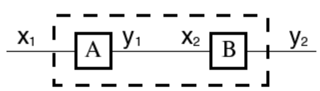
corresponds to the Link Table

x	n
0	a
1	b

where a and b are scalars, $a^2 + b^2 = 1$, and $|\mathbf{0}\rangle$ and $|\mathbf{1}\rangle$ represent the states corresponding to the logical values of **0** and **1**.

3.1. Equivalence to matrices/tensors

Returning to our simple two-component two-variable situation, if we substitute variables for the usual numerical case counts, and perform linking and reduction as before, it is easy to see that the result is the same as multiplying the corresponding matrices.



x_1	y_1	n_1
0	0	a_0
0	1	a_1
1	0	a_2
1	1	a_3

$\otimes$

x_2	y_2	n_2
0	0	b_0
0	1	b_1
1	0	b_2
1	1	b_3

=>

x_1	y_2	n
0	0	a_0b_0
0	1	a_0b_1
~~0~~	~~0~~	~~a_0b_2~~
~~0~~	~~1~~	~~a_0b_3~~
~~0~~	~~0~~	~~a_1b_0~~
~~0~~	~~1~~	~~a_1b_1~~
0	0	a_1b_2
0	1	a_1b_3
1	0	a_2b_0
1	1	a_2b_1
~~1~~	~~0~~	~~a_2b_2~~
~~1~~	~~1~~	~~a_2b_3~~
~~1~~	~~0~~	~~a_3b_0~~
~~1~~	~~1~~	~~a_3b_1~~
1	0	a_3b_2
1	1	a_3b_3

=>

x_1	y_2	n
0	0	$a_0b_0+a_1b_2$
0	1	$a_0b_1+a_1b_3$
1	0	$a_2b_0+a_3b_2$
1	1	$a_2b_1+a_3b_3$

$$\begin{bmatrix} a_0 & a_1 \\ a_2 & a_3 \end{bmatrix} \begin{bmatrix} b_0 & b_1 \\ b_2 & b_3 \end{bmatrix} \Rightarrow$$

$$\Rightarrow \begin{bmatrix} a_0b_0+a_1b_2 & a_0b_1+a_1b_3 \\ a_2b_0+a_3b_2 & a_2b_1+a_3b_3 \end{bmatrix}$$

More generally, Link Tables can represent higher-dimensional matrices (tensors) and operations upon them. The advantage of Link Theory is that it is a much simpler, more intuitive notation, and a more direct representation of the underlying *discrete* combinatorics of the physical situation. With case counts > 0, we can represent logic circuits, sets, etc., while case counts < 0 are necessary to fully represent quantum phenomena.

4. Summary and implications

We have shown how Link Theory, a simple accounting of possibilities, can be used to represent logic and idealized digital circuits, arithmetic problems, quantum experiments and much more. We also demonstrated the equivalence of Link Theory to matrices and tensor algebra.

In a subsequent paper, we will show how the same principles can be used to derive the core laws of quantum physics, and how *these laws are properties of abstract mathematics*, and are not derived from physical reality at all.

Link Theory is in fact a *general theory of structure*. Using Link Theory, we can show the equivalence and correspondence of theories in previously completely disparate realms, and *thus bring the theory and the power of each realm to the others*. In particular, using this formalism, we can unite quantum phenomena (and thus quantum computation) with classical computation and mathematical forms.

5. References and related work

Bastin, T., and Kilmister, C.W. (1995), *Combinatorial Physics*. World Scientific.

Etter, Thomas, and Noyes, H. Pierre (1998). Process, System, Causality, and Quantum Mechanics: A Psychoanalysis of Animal Faith. Stanford Linear Accelerator Center Publication 7890. [www.slac.stanford.edu/pubs/slacpubs/7000/slac-pub-7890.html]

Gershenfeld, Neil, and Chuang, Isaac (1997). Bulk Spin Resonance Quantum Computation. *Science*, vol 275, p. 350.

Leibniz, G. W. (1956). *Leibniz-Clarke Correspondence*. Alexander, H. G., (ed), Manchester University Press. See also Identity of Indiscernibles, http://www.bun.kyoto-u.ac.jp/~suchii/leibniz-clarke2.html.

Mackey, George W. (1963). *Mathematical Foundations of Quantum Mechanics*. W. A. Benjamin, New York.

Parker-Rhodes, Frederick (1981). *The Theory of Indistinguishables: A Search for Explanatory Principles Below the Level of Physics.* Springer.

Russell, Bertrand (1959). *My Philosophical Development*. London: Routledge.

Stein, Irving (1996). *The Concept of Object as Foundation of Physics.* Peter Lang.

Appendix A - Complex Case Counts

Complex case counts can be replaced by real case counts in Link Tables in a manner analogous to the replacement of complex values in quantum density matrices (Mackey, 1963). In the matrices, each value is replaced by a 4-component spinor, while in the Link Table, each row is replaced by 4 rows. Two additional binary variables (columns x and y) are also added enumerating phases 0-3 as shown below. Then the variables x and y are linked together serially in a chain, x to y to x to y, etc.

<u>matrix value</u>   <u>spinor</u>   <u>complex Link Table</u>   <u>real Link Table</u>

1   $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

v	n
•	1

v	x	y	n
•	0	0	1
•	0	1	0
•	1	0	0
•	1	1	1

-1 $\begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}$

v	n
•	-1

v	x	y	n
•	0	0	-1
•	0	1	0
•	1	0	0
•	1	1	-1

i $\begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$

v	n
•	i

v	x	y	n
•	0	0	0
•	0	1	1
•	1	0	-1
•	1	1	0

-*i* $\begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$

v	n
•	-i

v	x	y	n
•	0	0	0
•	0	1	-1
•	1	0	1
•	1	1	0

Appendix B - Quantum Pipeline Example

Result as a density matrix.

$$\begin{bmatrix} -1 & 1 & 1 & -1 \\ -1 & -1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

Result as a Link Table.

a	b	c	d	n
0	0	0	0	-1
0	1	0	0	1
1	0	0	0	1
1	1	0	0	-1
0	0	0	1	-1
0	1	0	1	-1
1	0	0	1	1
1	1	0	1	1
0	0	1	0	1
0	1	1	0	-1
1	0	1	0	1
1	1	1	0	-1
0	0	1	1	1
0	1	1	1	1
1	0	1	1	1
1	1	1	1	1