Structure Theory and the Quantum Core
[draft - including outtakes and unused raw material]

Tom Etter

Boundary Institute
(supported in part by Interval Research)
July 1999

CONTENTS

CHAPTER 1. Four Modes of Explanation

The history of physics has been a dialogue among four modes of description: *kinematics, dynamics, statics* and *atomics*, each with its own distinctive *mode of explanation*. The present essay will explore a more fundamental and more unified mode of description called *structure theory*, within which all four of these physical modes have their abstract precursors. Structure theory is a very general analytical tool. Surprisingly, certain of its general theorems turn out to provide a simple and natural explanation of the core of quantum mechanics.

First let's briefly review the familiar modes.

Kinematics is the study of motion as such. It reached its peak in the ancient world with Ptolemy, whose system of superposed epicycles became the explanatory mode of astronomy for almost two thousand years. The Copernican revolution, radical as it may have been in other ways, did not challenge this mode. In Copernicus' time, the epicycles of an orbit were regarded simply as its *parts*, just as C, E and G are the parts of a C-major triad. Why is a major triad made of notes in that ratio? Don't ask; that's just what a major triad *is*. Epicycles were the notes in the music of the spheres. Though they lost their prominence in astronomy after Kepler, their spirit remained alive through the centuries in the form of Fourier analysis, and regained center stage in our century with the Hilbert space formalization of quantum mechanics.

Dynamics, broadly speaking, is that mode of description that focuses on causes. When a cause is ongoing and quantitative, we call it a *force*, and it was Newtonian dynamics that attached this word to the product of mass and acceleration (whether this was a good idea is another matter.) Dynamics is the natural mode of technology, since a relationship of cause and effect is usually a potential relationship of means to ends. Dynamics represents process as the succession of states, and dynamical laws describe the dependence of later states on earlier ones. The goal of a dynamical analysis is to find laws which, when taken together with the initial state of a system, completely predict its future states. Notice how very different this is from the goal of analyzing a complex motion into a superposition of simpler motions. The two have one thing in common, however, which is their seeming finality. Once you have found a complete dynamical explanation, there seems to be nothing left to look for, just as in Ptolemaic kinematics there is nothing left to look for once you have found all the epicycles.

In around 1740 a curious digression occurred in the forward march of Newtonian dynamics when the French priest Maupertuis proposed that the laws of mechanics can be derived from what he called the principle of *least* *action*. Shortly thereafter, Euler and Lagrange put the principle of least action into a mathematically sound form, and Lagrange later proved that it was mathematically equivalent to Newton's laws. Thereafter it became merely a technical trick for analyzing dynamical systems. This demotion overlooked a profound distinction, however. Though least action allows the same *trajectories* as Newton's laws, i.e. it has the same kinematics, nothing could be more different as a mode of *explanation*. Causality is completely out of the picture, as in Ptolemaic astronomy. It's not how things *start* that determines where they go, but the inseparable fusion of *where* they start and *where* they *end up*.

There's another more subtle difference. The quantity called action, unlike Newton's force, has no obvious interpretation in everyday experience. What's more, it is not constrained by particular quantitative laws like Newton's laws governing force, velocity and acceleration, but only by the qualitative requirement that it be a function of position and velocity. Newton's quantitative laws come from assuming that action is a minimum; or more generally, an extremum. This raises an obvious but curiously neglected question: Might this quantity called action be more fundamental than either matter or motion? To frame the question more broadly, could it be that mechanics, and even causality itself, rest on the structure of a deeper and simpler level of nature?

This broader question is the focus of the present paper, and we'll return to it after a brief look at the other two explanatory other modes, statics and atomics.

Statics, which is about finding the stable states of things, is often allied with principles of least something or other. The stable state of an avalanche is at the bottom of the hill. The stable surface of a soap bubble or a body of water is that of minimum area. When you stir a mixture of black and white sand, its color stops changing when it reaches a uniform gray, which is the state of greatest uniformity (principles of least and greatest are really the same, since the greatest of something is the least of its opposite.) This last example brings us to atomics.

By *atomics* I mean the explanation of lawful behavior as the expected large-number behavior of very many small parts. It's a particularly powerful mode of explanation since it can provide precise quantitative knowledge without needing much knowledge of the details – indeed, the essence of its method is to *ignore* the details. Atomics has had great success with statics problems involving heat, pressure, usable energy etc. (despite its name, thermodynamics is mostly thermostatics), and we now realize that the stability of the everyday world is the large-number stability of average atomic motions.

The principle of least action has the flavor of a principle of statics. Looked at in this way, might it too come under the aegis of atomics? Could least action simply mean least *improbability*?

Under this hypothesis, the "atoms" of mechanical motion would of course not be objects but events. Suppose that such events were the tiny "fits and starts" of a fine-grained random walk with drift, whose average trajectory is the macroscopic motion we observe. There is then a natural interpretation of action as the negative logarithm of probability, and, if we imagine time as a spatial dimension, the trajectory of least action becomes the "state" of maximum entropy. We know from the theorems of Euler and Lagrange that the solution of the variational principle will have the general form of Newtonian mechanics, so the question then becomes how this interpretation of action connects mass, energy and force to space, time and probability.

The answer is easily found (see Chapter 5). First of all, mass is inverse dispersion rate, as in Brownian motion (Bohm [ref.] showed how this leads to a classical interpretation of the Heisenberg uncertainty principle.) Kinetic energy is $\frac{1}{2}mv^2$, where v is the average actual velocity. What is more interesting is that potential energy turns out to be $-\frac{1}{2}mw^2$, where w is the drift velocity of the random walk (the drift velocity will not in general be the actual velocity, since the final state of the trajectory is given independently of the initial state, which means that it need not be the expected state given the initial state.) If the tiny "fits and starts" are all at some constant velocity c, the kinematics become relativistic, and there is even a simple derivation of $e = mc^2$ that makes no use of kinematics at all.

What might have been the history of physics if these results had been discovered in the Eighteenth Century? The general idea of equilibrium as a state of maximum probability was certainly abroad then, thanks to Hook, Boyle et al. What was lacking was an adequate theory of probability. By the time that had arrived, the dynamical interpretation of Newtonian mechanics had become so entrenched that the above reasoning was probably unthinkable. It is thinkable today, but no longer plausible as an explanation of mechanics, since it doesn't lead to quantum mechanics. Also, it starts out naively with space-time, which today cannot be divorced from matter and must itself be explained rather than just taken for granted.

One can't always expect to hit the bull's eye with the first shot, however. What we learn from this example is that it makes good sense to try to explain the laws of mechanics as large-number laws. That's the lesson we'll apply in this paper. Our goal here, however, is not to explain all of mechanics – we shall, in fact, ignore space, time and energy altogether – but rather to gain a solid and clear understanding of what I'll call the *quantum core*.

What I mean by the quantum core is the part of quantum mechanics that follows from the generalized Born probability rule prob(P) = trace(PD) plus the generalized dynamical law UD' = DU, where P is a projection representing a proposition, D a density matrix, U a unitary transformation, and D' the transformed density matrix. This small part of physics turns out to be a natural unit that separates cleanly from the huge edifice of space-time-energy physics. It is not without interest in its own right, however. For one thing, it's where the essential "quantum weirdness" resides. It's also of a growing practical interest today, since it's exactly what's needed from physics for the logical design of quantum computers.

We'll see here that the quantum core laws can be interpreted as large-number phenomenon within a pre-physical "atomism" which also encompasses classical Markov chains. As a covering theory for both quantum and classical, our new framework also covers a wide range of processes that are neither quantum nor classical. Among these are certain quantum-classical hybrids that exhibit the standard rules of quantum measurement. But there are also many others we have not yet looked for in nature, some of which would be rather shocking to today's scientific common-sense if they turned out to exist.

So what, then, are our new atoms? Speaking very roughly, they are anything and everything. Speaking more accurately, they don't exist. Though our mode of explanation resembles atomics, it is really a new mode. Our large-number counts are not counts of anything in particular, but arise from the structure of *structure itself*.

CHAPTER 2. Structure, Shape and Congruence

2.1 Russell's relation-arithmetic

Russell and Whitehead had planned to cap off their *Principia Mathematica* with a last volume devoted to what they called *relation-arithmetic*, which was to be a general theory of mathematical structure. Alas, this intriguing project was never completed. The existing *Principia* does introduce its basic idea, but, as Russell tells us in his book *My Philosophical Development*, progress ground to a halt when they came to higher-order relations. "Whitehead was to have dealt with them in the fourth volume, but after he had done a lot of preliminary work, his interest flagged and he abandoned the enterprise for philosophy." [ref.] In fairness to Whitehead, it must be noted that Russell also abandoned the enterprise for philosophy.

.

Russell had a vision of relation-arithmetic as a tool which would extend the power of ordinary arithmetic to structure in general, including the structure of the empirical world.

> *"I think relation-arithmetic important, not only as an interesting generalization, but because it supplies a symbolic technique required for dealing with structure. It has seemed to me that those who are not familiar with mathematical logic find great difficulty in understanding what is meant by 'structure', and, owing to this difficulty, are apt to go astray in attempting to understand the empirical world. For this reason, if for no other, I am sorry that the theory of relation-arithmetic has been largely unnoticed." Bertrand Russell [ref.]*

Russell's vision will become our point of departure.

First, a few standard definitions to fix our terminology:

By a *relation* will be meant an open statement, i.e. a predicate, in a context of things to which it applies. We'll assume in this paper that this context is finite. By a *relationship* is meant an *instance* of a relation, i.e., a closed statement that results from substituting definite *values* for the free variables of a relation. Thus $x<y$ is a relation and $3<5$ a relationship which is an instance of $x<y$. The set of all values that occur in some instance of a relation is called the *range* of that relation. An instance of $x<y$ specifies an *ordered pair* of values which satisfies $x<y$, and the set of all such ordered pairs is called the *extension* of $x<y$. More generally, the *extension* of an n-term relation R is the set of all *ordered n-tuples* of values which create instances of R when their values are substituted for the free variables of R.

A predicate and its extension are of course very different kinds of things, but that difference is often overlooked in the mathematical treatment of relations, and in many works the two are simply equated. Since the concept of structure, as here presented, is abstracted from the concept of extension, we'll go along with this somewhat regrettable practice, using the terms "relation" and "extension" more or less interchangeably according to what needs to be emphasized. There are better ways to go, but they would take us too far outside the scope of the present essay.

The above definition of extension in terms of ordered n-tuples, though it is the standard one, is not very precise, and needs some fine tuning. Which value in an n-tuple should be substituted for which free variable? In practice, the i'th value is substituted for the i'th variable in some common expression for the predicate, e.g., "x<y". But this doesn't work very well for compound predicates like "x<y AND y<z", which could just as well be written "y<z AND x<y". In the first sentence the order is x, y, z, while in the second it's y, z, x. Which should we choose? To avoid such arbitrary choices that depend on grammatical accidents, it's better to represent the extension of a relation as a *table* of values whose columns correspond to the free variables of the relation, and whose rows represent the n-tuples that satisfy the relation. Truth tables, for instance, are extension-tables of Boolean relations. Here is the extension-table of x<y over the range of integers{1,2,3}.

x	y
1	2
1	3
2	3

Fig. 2.1

One can choose the order of columns in an extension-table to match the order of variables in one's favorite grammatical form of the table's predicate statement. But if we regard expressions like "x<y AND y<z" and "y<z AND x<y" as making the same statement, then we cannot think of the column order as belonging to extension itself. For any predicate P, the rules of the predicate calculus give us the means to construct a predicate P' which is logically equivalent to P, but whose variables are rearranged in any order we choose. The table of P' will then have a correspondingly different column order. If we regard P and P' as interchangeable, we have no choice but to allow the arbitrary reordering of columns.

Be aware that this constitutes a real change in the standard definition of extension; I hope the reader will agree it's a change for the better.

The standard definition says that an extension is a *set* of n-tuples. This means that two extension-tables represent the *same extension* if one can be transformed into the other by permuting the order of rows. According to our new definition, two extension-tables also represent the *same extension* if one can be transformed into the other by permuting the order of columns. We need a term that encompasses both kinds of transformation:

Table permutation: A permutation of the rows of a table, or of the columns of a table, or of both. We'll follow the usual convention of letting the word *permutation* mean either the transformation itself or the result of that transformation.

Extension is what it invariant under all permutations of an extension table. If we think of a table as an expression, its permutations are its grammatical variants. We can also think of a table as a function on a *grid*, where by a grid we mean the Cartesian product of two *indices*, a row index i and a column index j, and by an *index* we mean a set of contiguous integers starting with 1 (or 0, for a *0-based* index.)

Let's call the intersection of a row and a column of a grid a *grid cell*.

Grid cell: An ordered pair <i,j> of indices. If we think of a table as a function F(i,j) on the cells of a grid, a *table permutation* of F(i,j) is then another function F'(i,j) = F(i',j'), where i' is a permutation of i and j' a permutation of j.

The term *cell* also applies to tables, and we'll take that to be its default meaning.

Cell (table cell): An ordered triple <i,j,v>, where i is a row index, j a column index and v a value. We'll define *Row(c), Col(c)* and *Val(c)* to be the components of a cell c, i.e. if c = <i,j,v>, then Row(c) = i, Col(c) = j and Val(c) = v. Also, we'll define the expression *Row(c=d)* to mean that cell c is in the same row as cell d, *Col(c=d)* to mean that it's in the same column, and *Val(c=d)* to mean that it has the same value.

Table (formalized): A *table* T(i,j,v) is the *extension*, in the old-fashioned sense, of a three-term functional relation (F(i,j) = v) on two indices, i.e. it is the set of all ordered triples <i,j,v> such that F(i,j) = v. A *table permutation* of T is thus the set of all cells of the form T'(i,j,v') where v' = F(i',j') and i' = R(i) is a permutation of i and j' = C(j) is a permutation of j.

Having taken care of these relatively fussy matters of set-theoretic representation, we can now come to the real meat of Russell's relation-arithmetic, which is given by a definition that parallels that of cardinal number. The latter essentially dates from Cantor, and is based on the recognition that two sets have the *same cardinality* if and only if the members of the first can be replaced one-for-one by the members of the second. Russell and Whitehead followed Frege in applying the term *similar* to such sets, and then extended the same term to relations:

Similarity. Two relations R and R' will be called *similar* if we can transform the extension of R into the extension of R' by replacing the values of R one-for-one by the values of R'. More formally, table T is *similar* to table T' if there is a 1-1 mapping f(v) of the range of T onto the range of T' such that the table which is the set of all ordered triples of the form <r’,c’, f(v)> is a permutation of T'.

Here are some tables in the similarity class of the table in fig 1:

x	y
1	2
1	3
2	3

Fig. 2.2A

y	x
2	1
3	1
3	2

Fig. 2.2B

x	y
9	6
4	6
9	4

Fig. 2.2C

x	y
John	Jim
John	Mary
Jim	Mary

Fig. 2.2D

The relation-number of a table is that which it shares with all similar tables. Since relation-numbers are designed to be combined and operated on, it's important to define them as mathematical objects which we can point to. Russell and Whitehead again followed Frege and defined a relation-number as a *similarity class*. This is not a very good definition, though. Even in pure mathematics, such open-ended classes must be handled with kid gloves to avoid paradoxes, while in the empirical world they are usually nonsensical. Does the class of all relations similar to R refer to present relations only, or does it include past and future relations? How about *possible* relations? Already we are on swampy ground. A general theory of structure should not have to deal with such questions. Fortunately, there is an easy way to avoid them.

Structure table. Define a *structure table* as an extension-table whose range is an index.

A structure table is a function from two indices onto a third. The necessary and sufficient condition for two structure tables S and S' to represent similar extensions is that one can be gotten from the other by permuting all three indices. The class of all structure tables similar to S is thus an unproblematic finite set. Let's then point to such finite sets as the mathematical objects called *relation-numbers*. Given any table T, there is a structure table S similar to T . To produce S, simply number the values in the range of T and substitute the numbers for the values. It doesn't matter how you number them, since any other numbering gives a structure table that is a member of the same relation-number. Thus every table, and consequently every relation, has a unique relation-number.

Relation-number: A maximal set of similar structure tables. For reasons that will be become clear in the next section, we'll also speak of relation-numbers as *shapes*.

(footnote on the *canonical* table of a relation-number, which is analogous to a von Neumann counter. The idea is to interpret a relation-table as an integer whose digits are the value indices in base n, where n is the number of values. The canonical table is then the least such number in the relation-number of a relation-table.)

It should be noted, by the way, that the relation-numbers of one-place predicates can be identified with ordinary cardinal numbers, since such relation-numbers are uniquely characterized by the cardinalities of their ranges.

The above formal definitions constitute an "encoding" in the language of set theory of ideas that are for the most part quite commonplace and familiar. This encoding has much about it that is arbitrary, which not only makes it hard to understand but can in some situations be positively misleading. As mentioned above, there is a better way to go. This way avoids sets altogether, resting instead on the predicate calculus plus the theory of identity, which is expanded to include three equality predicates instead of just one. To get a hint of how this works, let's take a brief glance at another way to encode relation-numbers in set theory based on the three equivalent relations above that were called Row(c=d), Col(c=d) and Val(c=d):

RCV relation-number: Define an RCV structure table as an indexed set whose elements are called *cells* on which are given three equivalence relations, Row(c=d), Col(c=d) and Val(c=d), such that Row and Col are independent and Val is functional in Row and Col, i.e., Row(c=d) and Col(c=d) implies Val(c=d). Two RCV structure tables are called *similar* if they differ by a permutation of the cell set, and an RCV relation-number is defined as a maximal set of similar RCV structure tables. It's easy to show that two tables have the same RCV relation-number iff they have the same relation-number as defined above.

We'll not take this path here, and indeed for most purposes we'll ignore our set formalization altogether, treating relation-numbers simply as the things that are *referred to* by structure tables, where similar structure tables are to be regarded as grammatical variants of the *same expression*, just as "B and A" is a grammatical variant of "A and B". As long as our results don't depend on the order of rows and columns, and obviously remain true under any permutation of values, this practice will not lead us astray.

Tables are not the only way to represent extensions, and there is an alternative to structure tables that will turn out to be very important in our analysis of quantum mechanics. Given an n-column structure table T with m values, create an n-dimensional array of 0's and 1's in which each dimension corresponds to a column of T and is indexed by the value index of T. The cells of this array correspond to all possible rows of an n-column m-row structure table, and the 1 and 0 cell values represent the presence or absence of the corresponding row in T. Let's see how this works for the two-column three-valued table of fig 2.1 (notice that it is in fact a structure table.)

Fig 2.1.1 Table Matrix

The bit-array of a two-place relation is of course a square matrix. It is such matrices, and their generalizations to partial tables that connect relation theory to the linear operators of quantum mechanics. To get a hint of why this is so, note that there is a theorem to the effect that multiplying the bit-arrays of functional relations corresponds to composing their functions. We'll see in Section 2.3 how this theorem extends to binary relations in general.

As mentioned, Russell hoped that relation-arithmetic would become a general theory of mathematical structure. It should be noted that his concept of similarity is closely related to the familiar mathematical concept of *isomorphism*. Let G and G' be two groups. To say that G and G' are isomorphic means that there is a one-one correspondence between G and G' that preserves group multiplication. Another way to say this is that the three-term relation x = yz on G is similar to the three-term relation x'=y'z' on G'. Thus the *abstract group* G can be identified with the relation-number of the relation x = yz.. This works for algebraic structures in general; for instance, an abstract field is the relation-number of a six-term relation of the form (x=y+z AND x'=y'z'). The ability to freely combine and take apart relation-numbers should thus be a powerful tool for unifying algebra.

So why wasn't this tool ever developed? What went wrong?

2.2) Relational congruence

The problem, in a nutshell, is that relation-numbers *don't combine*! To combine A and B, we must first bring them into the same universe of discourse, i.e. we must be able to say "A and B". The trouble is that, for predicates A and B, the relation-number of the predicate (A AND B) is not a function of the relation-number of A and the relation-number of B. Conjunction is *not invariant* under similarity. This non-invariance is fatal for relation-arithmetic as Russell and Whitehead conceived it. It's not clear whether they were aware of this problem, but we do know that they "abandoned the enterprise for philosophy."

Suppose predicates A and B have no free variables in common. Let IA be any instance of A, and IB be any instance of B. Then (IA AND IB) is an instance of (A AND B), and of course all instances of (A AND B) are of the form (IA AND IB). Thus the extension-table of a conjunction is formed by combining the columns of A and B, and taking each pair of rows from A and B as a row. In database terminology, this table operation is called the *Cartesian product,* or *cross product*, written A⊗B. Since “Cartesian product” would have a different meaning for tables regarded as cell sets, we’ll use “cross product”. Here are two very simple examples:

X	y
0	1
1	0

⊗

x'	y'
0	1
1	0

=

x	y	x'	y
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1

Fig. 2.3A

x	y
0	1
1	0

⊗

x'	y'
2	1
1	2

=

x	y	x'	y
0	1	2	2
0	1	1	1
1	0	2	2
1	0	1	1

Fig. 2.3B

Fig 2.3 The non-invariance of the cross product

Notice that, though all the components are similar, the cross product in 2.3A is not similar to the cross product in 2.3B. The relation number of the cross product is thus not a function of the relation-numbers of its components. This was what was meant above by the statement that conjunction is not invariant under similarity.

A less trivial example of the failure of invariance is provided by the six-term relation above, whose table is the cross product of the tables x = y+z and the table of x' = y'z'. These two components are independent in their product table (this is because any true statement of the form a= b+c is consistent with any true statement d = ef, where a, b, c, d, e, and f are constants). Nevertheless, they are intimately tied together by having a common range of values, the nature of that tie being defined by the distributive law. Given only the relation-numbers of x = y+z and x' = y'z' separately, there are no grounds for even deducing that their tables have a common range, much less that they combine into a table with the relation-number of a field. Yet the ability to compose and decompose structures like fields is just the kind of thing we should hope for from a general theory of structure.

The most important operation on relations is composition. The failure of invariance for cross product means that composition is not invariant either. A theory of structure that can't even deal with relational and functional composition would not seem to have much of a future.

Fortunately, the invariance problem has a fix. We don't have to abandon relation-arithmetic for philosophy after all. We’ll now see that it’s possible to retain the requirement of invariance provided we supplement the equivalence relation of similarity with a second more "social" equivalence relation called *congruence*. Relational congruence applies to the *parts* of a relation, just as geometric congruence applies to the parts of a space. Congruence is itself invariant under similarity, so this new step does not change Russell's basic conception of structure. What it does change is his conception of the part-whole relationship as it applies to structure.

Euclidean geometric figures are called *similar* if they have the same shape. The *shape* of an object is thus its *similarity class*, to borrow Russell's terminology. Since Euclidean geometry has no intrinsic unit of distance, shape does not include size. This means that shapes aren't building blocks. You can't build anything out of shapes, not even other shapes.

That in a nutshell is where Russell's relation-arithmetic went wrong: it tried to build shapes out of shapes.

Shape. From now on this is the word we'll use for what Russell called *relation-number*. That is, *shape* will be used to mean what is invariant under similarity, just as it does in geometry.

If you want to lay a hexagonal tile floor, you don't just order hexagonal tiles, you order *congruent* hexagonal tiles. If all you care about is that they fit together, you don't have to specify their actual sizes, but you do have to specify that they be of the *same size*. Now being of the same size is invariant under similarity, so ordering congruent tiles doesn't really take you out of the realm of pure shape. It does, however, restrict you to ordering tiles from suppliers who live in the *same space*.

If you want to construct a "tiled" table like that of fig 2.3A, you can't just put together parts of the same shape, since that could just as well give you fig 2.3B. Rather, you have to put together *copies* of the same relation. If all you care about is how they fit together, you don't have to specify the actual values in these copies, but you do have to specify that they have the *same values* in the *same places* (modulo a grid permutation). Now sameness of value, within a single relation table, is *invariant under similarity*, so ordering such value-for-value copies doesn't really take us out of the realm of pure shape. It does, however, restrict us to ordering our copies from suppliers who live in the *same shape*..

Geometric congruence was defined above as having the same shape and size. Since size is literally meaningless in Euclidean geometry, the following is a better definition: Figures A and B are called congruent if A can be mapped onto B by a combination of translations, rotations and reflections, i.e., by a transformation belonging to the so-called *Euclidean group*. We'll define relational congruence in a similar way, where the analogue of the Euclidean group is the group of grid permutations, and the analogue of a figure (point-set) is a cell-set, i.e. a subset of the cells of a table.

Relational congruence. Let C and C' be cell-sets of the a relation-table T. We say that C and C' are *congruent* in T if there is a grid permutation G that maps the cells of C onto the cells of C' in such a way that corresponding cells have the same value. The permutation G will be called a congruence permutation.

Theorem. Relational congruence is invariant under similarity.

> Proof: A similarity mapping is the product of a permutation and a substitution of values. Clearly congruence is unchanged by a substitution of values, since we only require that corresponding cells have the *same* value. Let P be any permutation of T, and let G be a congruence permutation that maps C onto C' . Then GP^{-1} is a congruence permutation that maps P(C) onto P(C') in P(T). QED

The only cell-sets we'll be concerned with here are those that are tables in their own right:

Sub-table. A sub-table of a table T is define as a cell-set that is the intersection of the cells in a set of rows of T and the cells in a set of columns of T. T is of course a sub-table of itself.

Our program now is to reformulate Russell's relation-arithmetic so that it applies to a new kind of "relation-numbers" that are invariant under congruence rather than under similarity. Though every cell-set has a relation-number in this new sense, our new arithmetic will only apply to sub-tables, and we'll speak of their numbers in particular as *congruence-numbers*. How, then, do we point to a particular congruence-number? Recall that we pointed to a shape as an equivalence class of similar structure tables. There is no direct analogue of this definition for the congruence relation, however. We must proceed in two steps. First, let's see how to define a *representation* of a congruence-number in a table:

Congruence class: Given a table T and a sub-table S of T, the congruence class of S is the set of all sub-tables congruent to S. To put it another way, a congruence-class of a table is an equivalence class of sub-tables under congruence.

Congruence classes are not congruence-numbers, since they belong to tables rather than shapes. By restricting these tables to structure tables, we take a step in the right direction, but there is still the problem of "factoring out" the arbitrary ordering of the three indices of a structure table. What we need is a definition that represents a particular congruence-number as a unique mathematical entity.

Structure tables T and T' represent the same shape if T' can be derived from T by permuting all three of its indices. Let's call such a permutation a *shape permutation*. A shape is thus a particular table T taken together with all other tables T' that are shape permutations of T. Now each T' is generated by a unique shape permutation C'(T). Since congruence is invariant under similarity, a congruence class K of T is mapped by C' onto a congruence class K' of T'. Let us then call the set whose members are K together with all of the K' a *congruence-number*.

Congruence-number: Let S be a shape. Then a congruence-number N of S is defined as the set of all shape permutations of a congruence class of some table belonging to S.

2.3) Duplicate rows, count tables and characteristic tables

There is one very important way in which a sub-table may differ from the table of a relation, which is that the same row may occur in it more than once. Since the rows of a table are all *instances* of the same relation, it might seem that this is redundant information - after all, "2<3 and 2<3" is just a long way of saying "2<3". This is true for the table as a whole. However, by removing duplicate rows from the sub-tables we may discard crucial information about which are *congruent* to which. Duplicate rows in sub-tables, far from being redundant, *belong to the concept of congruence* and thus to the very concept of shape.

To repeat: The congruence-number of a part includes its multiplicities. As we'll see in the next section, this is the key to understanding how quantum probabilities and quantum amplitudes both reside in the concept of shape.

Though we do have to count the duplicate rows, we don't have to write down each duplicate as a separate line; it's quite sufficient to write down each *distinct* row followed by its *count*. This is more than a theoretical consideration when the counts get into the millions and billions, as they easily can with more than a few parts.

Count table: An abbreviation of a partial table that omits duplicate rows but has an additional *count* column showing the number of occurrences in the original. Remember, count tables are only *abbreviations*, so whatever we say about them is really about partial tables as defined above. We won't insist that all lines of a count table be distinct, since it is sometimes useful to divide a group of identical rows into several parts. Indeed, a count table needn't "count" at all; merely adding a count column with all 1's to a relation table turns it into a legal count table.

Count tables are not as esoteric as they might at first seem. Take the relation "x is the father of y". Let's make a table of this relation in our neighborhood. It would contain rows like "Jim, Suzie", "Jim, Mary", "John, Meg", John, Bill", "John, Jason" etc. Now select the partial table consisting of the first column alone. This table contains Jim twice, John three time etc.. Its *count table* is thus a list of facts of the form "Jim is the father of two", "John is the father of three" etc. To say that John is a father of three says more than that he is a father, but less than that he is the father of Meg, Johnny and Jason. It's this *more* that situates fatherhood in the encompassing relation of father-child, and in the even more encompassing relation of father-mother-child. It's this *more* that transforms the status of fatherhood from that of an *aspect* of father-child to that of a *component* of father-child. Most important, it's this *more* that will turn out to be the basis of structural atomism.

x	y
Jim	Suzie
Jim	Mary
John	Meg
John	Billy
John	Jason

x
Jim
Jim
John
John
John

x	n
Jim	2
John	3

Fig 2.4

It is often useful to extend a count table by adding rows with count 0. Such rows merely announce that they are not there, so-to-speak. Since we can discover this information by looking at the rows that are there, such so-called *empty* rows don't change the reference of a count table. However, they are an important formal tool, as we'll see in the next two chapters. One way in which they are useful is in "padding" a table so as to include all possible rows. The count column of such a padded table then represents the *characteristic function* of the subset of rows that are in the extension table. More exactly:

Characteristic table. A count table whose counts are either 0 or 1, and which contains all possible rows, given the values of the relation.

To construct the characteristic table of an n-term relation, form a single column table with all of its values, take its n-th power as a cross-product, and then write the characteristic function of the extension of that relation as a last column.

Complete count table. We can also add 0-count rows to a count table whose other rows have counts greater than 1. If we thereby include all missing rows, the resulting count table will be called *complete*.

Characteristic tables and complete count tables have crucial roles in bridging the conceptual gap between structure theory and standard quantum mechanics. Recall that in Section 2.1 we encountered the bit matrix representation of two-column tables. Such a representation is a reordering of the information in a characteristic table, as defined above. Complete count tables can be reordered in exactly the same way, which leads to a generalization of the product theorem for functional relations we encountered in Section 2.1. The broader theorem says that multiplying count matrices corresponds to composing their partial relations. We'll see examples of this in Section 3.1, and the mathematics will be laid out rigorously in Chapter 5. It is the (normalized) counts in such count matrices that will be interpreted as both amplitudes and probabilities in our derivation of the quantum core.

2.4) Structure theory

Though the present essay is about *structure theory*, we have still not brought on the main character. What, then, is a structure?

The words *structure,* s*hape* and *form* are often used interchangeably, but their roots in ordinary language are quite different. Form is the polar opposite of *matter*; it's how an object has been *formed* or *shaped*, in contrast to what it is made of. Structure, on the other hand, means the arrangement of the parts in a whole, and is usually contrasted with the bare collection of these parts. A vase turned on the potter's wheel has a shape, in contrast to clay as such, which is shapeless. A house, on the other hand, has a structure, in contrast to the pile of already formed bricks and boards that were put together to make it. In more abstract terms, form is *shape*, structure is *composition*.

That's how we'll use the word here. Structures will not be identified with shapes, either complete or partial. Rather, a structure is to be understood as a shape that has been taken apart and put back together again, so-to-speak. To put it another way, a structure is a shape presented as the composition of its parts.

.

We have met one kind of structure in the structure table. Its "bricks" are its cells, and the arrangement of these bricks is the rectangular order defined by the two indices c and r. Of course, to see a structure table *as* a shape rather than *as* a particular table requires that we understand what to ignore in its table structure, which is almost all of it. To assist in this understanding, we constructed a much bigger and busier structure consisting of all the other structure tables that are similar to our table. We even went so far as to say that this big structure *is* the shape. That was not a very accurate statement. The big structure is not the shape itself but a tool for seeing the shape, and this tool shows us nothing unless we understand how to use it.

It was mentioned above that a good theory of structure should be able to analyze an algebraic structure like a field into parts called addition and multiplication. We saw that this can't be done if we take those parts to be relation-numbers, in Russell's sense. Can it be done with our new kind of parts? The answer is in fact yes, though we'll postpone the proof until we have the right tools for the task.

In this chapter we've focussed on shape, since the formal study of structure goes well beyond the agenda of Russell's relation-arithmetic.. But before moving on, a few general words are in order about structure theory as a mode of description and explanation, and how it relates to the four empirical modes we met in the first section.

Structure theory is, in effect, a fusion of abstract statics, kinematics, dynamics and atomics. Statics, in the broadest sense, is that mode of inquiry that treats its subject matter as entirely present. This fits the concept of shape, as defined above. Kinematics and dynamics, on the other hand, address a changing present; they are concerned with transformations, selections, fusions, partitions, connections, creations, destructions and shifting viewpoints. Dynamics is sometimes in the service of statics, as it was when we "transformed away" the index structure of a table in order to arrive at its shape. Structure is partially dynamic, since a construction or a composition must be understood as the result of a series of *acts* of constructing or composing. It's also partially static, since structure itself is not what changes.

Finally, atomics enters our present scheme in the form of count tables, which are the building blocks of shape. It comes into play when the counts themselves, rather than the particular things counted, become our objects of study,. We'll see in Chapters 4 through 6 how counting rows in the parts of a structure, any structure, results in a unified conception of probability and amplitude. The basic step that reveals this unity is essentially dynamic, since it involves the steps that *produce* the expansion and contraction of what is present. When a horizontal part is expanded in a way that disconnects its parts, it produces *count matrices* that behave like generalized quantum density matrices. From these we derive amplitudes. If we reverse this operation by reconnecting the disconnected parts, these density matrices turn into probability vectors that are quadratic in amplitudes.

When the density matrices are symmetrical in rows and columns, we get the quantum core. Classical Markovian processes are defined by another kind of density matrix symmetry, which is that the amplitudes at one of the broken ends of any connection are uniform. Connection laws in general have the same algebraic form as the quantum core, although the density matrices and transformation matrices needn't have these special symmetrical forms. What's important to realize is that this generalized quantum core rests on no assumptions other than those that go into the definition of shape itself. Quantum and classical are small islands within the vast ocean of structure, distinguished only by their characteristic large-number symmetries.

CHAPTER 3. Structure and Timeless Change

In the last chapter we approached structure theory very abstractly and ended up, not with a definition of structure, but with definitions of shape and type. In this chapter we'll proceed more concretely, working largely with examples of structure. Though we won't arrive at a formal definition of this elusive concept, our examples will lead us to a better understanding of how structure is related to the process of construction. In particular, we'll see how to make the steps of any construction commutative, thus separating the "dynamics" of structure from temporal succession. In Chapter 5 we'll turn to the complementary study of the "statics" of structure as something intrinsic to shape.

3.1) Equations, clauses and compounds.

Let's consider the familiar process of solving equations.

An equation says that one functional expression is equal to another. Take $2x = y+3$, which says that the function *twice* applied to x is equal to the function *plus 3* applied to y. Equations are *open* statements, which is to say, they contain unassigned or *free* variables. An open statement in itself is neither true nor false, but becomes true or false when its variables have all either been assigned to constants or bound by quantifiers. If we are given a definite range of values for its variables, then an equation becomes a *relation,* as defined in Chapter 2.

A *solution* of an equation is an assignment of definite values to its free variables that makes it true. Thus assigning 5 to x and 7 to y is a solution to $2x = y+3$. When solutions are defined in this way, it makes sense to speak of *solutions* of open statements in general; for instance, assigning 3 to x is a solution of $x<5$. Solutions, in this broader sense, are synonymous with *instances* of a relation. Thus a *solution table* is an *extension table*, as defined in Chapter 2.

We can always rewrite an open statement in the form of an equation. Consider $x<5$. Let $f(x)$ be the function which is 1 if x is less than 5, otherwise 0. Then the statements $f(x) = 1$ and $x<5$ both say the same thing about x, so they have the same extension table. Since extensions are our subject matter here, open statements and equations can be regarded as interchangeable.

Consider now the two simultaneous equations $y=x-2$ and $x=2y$. For simplicity, we'll confine them to the range of integers 1 through 4. One way to solve them is to substitute 2y for x in the first and solve the resulting single equation in y, substituting that value back into the second equation to find x.

Relation tables (i.e., extension tables) give us a conceptually simpler and more orderly procedure. Having several equations means we have already broken up a complex open statement into simpler clauses. Our new method assumes that these simpler clauses are individually soluble, which is certainly true in the present case.

The first step is to *separate* the equations by replacing their variables in such a way that no two equations have a variable in common. For our example, we'll replace x and y in x = 2y by x' and y'. Let T be the table of y=x-2 and U be the table of x' = 2y'. The modified equations are independent, which means that the cross-product C = T ⊗ U is the table of their conjunction. C has four columns, call them x, y, x', y'. To solve the two original equations simultaneously requires that we put the modified equations back together, which we do by adding two more equations, the so-called link clauses x=x' and y=y'. This means removing all rows from table C in which x ≠ x', and then removing all rows in which y ≠ y'. Finally, to obtain the solution table proper we must remove the redundant columns x' and y'. Here are the tables produced by this sequence of operations:

T

x	y
3	1
4	2

y=x-2

U

x'	y'
2	1
4	2

x'=2y'

T⊗U

x	y	x'	y'
3	1	2	1
3	1	4	2
4	2	2	1
4	2	4	2

(y=x-2) & (x'= 2y')

x	y	x'	y'
3	1	2	1
4	2	4	2

…& (y=y')

x	y	x'	y'
4	2	4	2

…& (x=x')

x	y
4	2

solved

Fig 3.1A.1 Constructing a solution table of two simultaneous equations

At the beginning of Chapter 2 we defined a *relation* to be an open statement in a context that gives it a definite extension. A set of simultaneous equations is a relation whose open statement is a *compound* of *clauses* which are equations, and whose *context* is the set of things which the equations equate, e.g. numbers.

Clause: A sentence which is not distinguished from logically equivalent sentences, where two sentences are called *logically equivalent* if they have the same free variables and are true for the same assignments to these free variables.

For instance, (A & B) is the same clause as ~(~ A v ~ B), and, in our previous example above, x<5 is the same clause as f(x) = 1. "Clause" is really another term for *extension*, defined in terms of sentences rather than tables.

Compound: A *conjunction* of clauses, called its *components*. Two sentences, whatever their grammatical forms, are regarded as the same compound if and only if they contain the same clauses. The clauses of a compound will usually be marked by enclosing them in parentheses.

A clause is a *logical* entity, while a compound is also a *grammatical* entity. Thus the sentence (A) & (B), as a compound, has two components while the logically equivalent sentence ~(~ A v ~ B), as a compound, has only one component. The clause/compound distinction is very useful in describing hierarchies, since we can turn any compound into a component of another compound simply by calling it a clause.

Our first step above was to separate the variables of the clauses (y=x-2) and (x=2y), which transforms their compound into (y=x-2) & (x'=2y'). Since its clauses are independent, its table is the cross-product of the tables of its clauses. Our next step was to re-identify the two separated variables by adding the clauses (y=y') and (x=x'). These are *conditions* on the existing variables that reduce the number of rows in the table down to those satisfying our original equations. Notice that the compound (y=x-2) & (x'=2y') & (y=y') & (x = x'). is not logically equivalent to (y=x-2) & (x=2y), since it has four free variables instead of two. It's table gives the solution twice, once for x and y and once for their duplicates x' and y'. In Fig. 3.1 we simply erased the duplicate part of the solution. This is not just turning a blind eye, though, since the same thing can be done *logically* by existentially quantifying the variables x' and y'. That is, our final solution table represents the extension of the statement:

$(\exists x',y')\ ((y=x-2)\ \&\ (x'= 2y')\ \&(\ y=y')\ \&(\ x=x'))$.

Clearly this method for "solving" compound statements is a general one. Given any compound statement whose clauses can be solved individually, we first separate their clauses, next write down the tables of these separated clauses, next form the cross-product of these tables, next link the separated variables by erasing the rows in which they disagree, and finally, erase the redundant duplicate columns. The result is the extension table of the compound. Whether this is a practical project is another matter; when the component tables are large, or when the compound contains many components, it clearly is not. However, one often encounters logical puzzles which are hard, not because they are complicated, but because they are confusing, and here the method can be genuinely useful. It was used by Dick Shoup, for instance, to solve the classic AI problem SEND MORE MONEY (ref.)

Solving problems is not our agenda here, however. What makes the above method of separating and linking clauses important for our present purposes is that it already exhibits the algebraic form of the von Neumann core quantum laws. Let's take a brief informal look at why this is so.

3.1A The core laws, first pass

The core laws involve three kinds of matrices: density matrices, projection matrices and transformation matrices. Let's consider these in order:

Von Neumann discovered that quantum mechanical states, both pure and mixed, can be represented by so-called *density operators* on a Hilbert space. When we make a measurement, the measuring device "chooses" that basis in Hilbert space whose vectors correspond to the possible outcomes of this measurement. What makes the density operator such a powerful concept is that the *probabilities* of these possible outcomes are simply the *diagonal elements* of the density operator's matrix in this *measurement basis*.

Von Neumann also realized that *propositions* about a certain measurement are in natural correspondence with those projection operators that are diagonal in the measurement's basis. Let D be a density matrix and S be a statement about its measured variable, for instance "x>5". For every such S there is a diagonal projection matrix P whose 1's correspond to the values of x for which S is true. The diagonal of PD will thus contain the probabilities in D for these values, and 0 for the rest. Thus the probability of S is the sum of the diagonal of PD, which is the first core law, i.e.:

First core law: prob(S) = trace(PD)

Trace is usually defined as the sum of a matrix diagonal. This sum, however, is invariant under all linear transformations, i.e., trace(M) = trace(TMT^{-1}) for any (nonsingular) T. Thus the trace isn't really a property of matrices but of *linear operators*. In quantum mechanics, the first law is assumed to be true for any (ortho) projection P and any (self-adjoint) density operator D. But since we can always find a basis that diagonalizes P, restricting the first core law to diagonal projection matrices is not really a restriction at all.

Because D is self-adjoint, we can always diagonalize it too (that's the spectral theorem). This produces the "classical slant" on the *state*, where the probability distribution on the measured outcomes is the whole matrix, and pure states are *sharp*, i.e. their probabilities are either 0 or 1. If we want both D and P to be diagonal in the matrix statement of the first law, we must explicitly include in our statement a coordinate transformation that relates the two bases in which D and P are diagonal, which we can do by writing either prob(S) = trace($P(T^{-1}DT)$) or prob(S) = trace($(UPU^{-1})D$) for the appropriate linear transformation T or U. Trace has what for us is another very important property, which is that trace(AB) = trace(BA). It follows that trace($P(T^{-1}DT)$) = trace($(PT^{-1}D)T$) = trace($T(PT^{-1}D)$) = trace($(TPT^{-1})D$), from which we conclude that U = T. This brings us to our third kind of matrix, the *transformation matrix* T.

Which in turn brings us to dynamics. The fact that T = U means that, as far as observation goes, any linear change in the *state* of a system is equivalent to an inverse change in the *viewpoint* from which we observe that system. Hamilton in the Nineteenth Century discovered an analogous inverse relationship between objective and subjective change in classical mechanics. It's worth digressing for a moment into the history here, since it helps clarify the status of the quantum core laws within physics.

Hamilton discovered a certain fundamental class of coordinate changes – he called them *canonical* transformations – which can be regarded as the most general changes of viewpoint on a mechanical system that preserve its essential structure. He then proved a remarkable theorem, which is that a Newtonian evolution of state is simply the steady unfolding of a differential canonical transformation. An obvious special case of this theorem is the relativity of uniform motion, since the uniform motion of an object can't be distinguished from the opposite motion of the observer. But it is far from obvious that every Newtonian change, no matter how complicated, would be indistinguishable from an "opposite" change of viewpoint on an unchanging object, and indeed this feature of Newtonian mechanics has no counterpart in everyday experience.

It does have a simple counterpart in quantum mechanics, though. We saw above that the first core law, which is the basis of all quantum-mechanical *manifestation*, can be expressed in the coordinate-free language of linear operators, and is thus invariant under linear transformation. We also saw that when we apply T inverse to the state operator, we get the same change of probabilities we would get from applying T to the projection operators that correspond to propositions. If we assume Hamilton's principle of subject-object symmetry for mechanical change, and if we think of $P' = TPT^{-1}$ as the manifestation of a canonical transformation T, then we would have to describe the law governing a mechanical change of state by $D' = T^{-1}DT$. And indeed, for unitary T, this does characterize the Schrodinger equation and its generalizations. Multiplying both sides by T gives TD' = DT, which is slightly more general in that it also makes sense in case T has no inverse.

Second core law: TD' = DT, where D is the earlier state, D' the later state, and T a linear transformation.

It should be noted in passing that the formalism of so-called *quantities* as self-adjoint operators is derivable from these two core laws. In particular, one can prove that the average of a quantity Q in a state D is given by trace(QD), which is the rule by which the quantum domain usually manifests itself in the macroscopic world.

Let's now see how the core laws turn up in clause theory. At this point we'll be painting with very broad strokes; the mathematics will be spelled out in later sections.

We saw in Section 2.1 that the information in a two-column extension table can always be rearranged as a square matrix of 0's and 1's indexed by the values of the relation, where a 0 means that that pair of values is not in the table and a 1 means that it is. For instance, the matrix of a link x=y is the identity matrix, since the table of x=y only contains rows with equal values. The numbers 0 and 1 can be regarded as *counts* of the number of copies of a row in the table. Thus we can extend this matrix construction to the count tables of all two-place partial relations.

In our example of the two simultaneous equations in Section 3.1, we created a compound in which the two equations are introduced as independent clauses and then linked by two link clauses. Let's limit ourselves for now to compounds in which the non-link clauses have no variables in common.

Any two-place clause, considered separately, has its individual count matrix. It can be shown, as mentioned in Chapter 2, that when we *link* two such clauses and hide their link, we *multiply* their individual matrices.

Any two-place clause in a compound statement S also has its count matrix as a part of S, which we get by hiding all the other variables. Indeed, the same is true for any pair of variables x, y in S, whether they belong to the same clause or not.. The count entry at cell x=i, y=j of that matrix is the number of solutions of S for which x=i and y=j. Now we come to the key definitions:

Density count matrix of a link: The count matrix of the *broken link*, i.e., the count matrix of the pair of variables x, y in the statement that results from removing the link x=y from S.

Density matrix of a link: The density count matrix normalized by dividing by its trace.

Probability: Define the *probability* that x has value k as the number of solutions in which x=k divided by the total number of solutions – in Pascal's terms, it's the ratio of favorable cases to total cases.

Now it's easy to see that the trace of any density count matrix is the total solution count, so, according to the above definition, the diagonal elements of the density matrix are the probabilities of the values of the link variable. Recall that this is just what characterizes the von Neumann density matrix, and what makes it so useful! We have in effect derived the first core law.

To spell it out a bit further, note first that by the probability of a value k, we mean the probability of the clause (x=k). Now modify the link clause (x=y) by adding x=k, giving the clause (x=y & x=k). Its matrix is a diagonal projection, call it P, with a single 1 at cell k,k. Attaching the link at x gives the matrix PD. Now the trace of PD is the equal to the trace of PD with its off-diagonal elements removed, which is what we get when we link y. In other words, the probability of x=k is trace(PD). The same is clearly true when we replace x=k by any statement S(x) about x, which proves:

First core law of clause theory. Given a compound C, a link x=y of C, and any statement S(x), we have prob(S) = trace(PD), where D is the density matrix of link (x=y) and P is the projection matrix of S, as defined above.

The concept of a transformation arises in clause theory when we are dealing with a compound C that has two parts PT and PU which become independent when we remove two links (t=u) and (t'=u') between them, where t and t' are in PT, u and u' in PU. Let D be the density matrix of (t=u) and D' be the density matrix of (t'=u'). The question is: What is the relationship between D and D'?

Consider the count matrix on the variables t and t' of part PT, with the links removed, and similarly on PU; we'll refer to these matrices as T and U. Then by the matrix product rule for linked matrices, D = TU and D' = UT. It follows that DT = TUT and TD' = TUT, hence TD' = DT, which is the second core law. The same argument shows that D'U = UD. Actually, the order of matrix multiplication depends on the arbitrary choice of which of t and t' is the horizontal and which the vertical matrix index, but we can always choose them to give these equations

Second core law of clause theory. In a compound consisting of two doubly-linked parts with link matrices D and D' and isolated matrices T and U, we have TD' = DT and D'U = UD.

If T or U have inverses, which they actually do for most partial relations, then D and D' become functions of each other, and the second core law can be written in the more familiar form D' = T^{-1}DT. By dividing T up into smaller parts, we can represent an ongoing process, which approximates a continuous process when the component T matrices are close to the identity.

These "clause theory" core laws, though they have the algebraic form of the quantum core laws, don't encompass quantum mechanics itself, since a unitary T will have negative entries unless it is just a permutation matrix. We'll see in the sections ahead how to make sense of such negative entries. However, the derivation above does get to the heart of the algebra. The crucial step was to define the "state" of a variable in terms of the relational structure of a *disconnection* that splits that variable in two; we'll explore this notion of state in greater depth in Chapter 4.

Whatever happened to Hilbert space and abstract linear operators? you ask. The simple answer is that, in our new relational scheme, their job is done by the linear invariance of the trace.

3.2) Constructing shapes (rewrite beginning)

Returning to our example in Section 3.1 of the two simultaneous equations, let's note something a bit odd. Our construction started with the "component" equation y=x-2 and then conjoined three more component equations; the table of the resulting conjunction is our solution table. The odd thing is that, although the first two equations *added* cells to our table, the second two *took cells away* from it. This shoots down the overly simple metaphor of equations as building blocks. Some of our "bricks" do in fact become parts of the structure, but others make holes in it. To look ahead a bit, the oddity of bricks making holes (maybe they should be called brickbats) will turn out to be the key to understanding negative counts and quantum superposition.

Second, we have naively brought in our equations as "separate" objects, without giving thought as to what it means for our parts to be separate. But it was just such inattention that sank Russell's relation-arithmetic, as we saw in Section 2. Being separate is *being related in a certain way*, and there are in fact many different kinds of separateness. Parts one and two above are separate in the sense of being independent. As we saw, independence says nothing about the relationship between the *ranges* of relational parts, which can vary from being disjoint to being identical (being disjoint is another kind of separateness). That's why Russell's relation-arithmetic couldn't handle composition, and why we need congruence. Parts three and four, though they might in some sense be regarded as independently *applied* constraints, are highly correlated with parts one and two in determining the extension of the composite relation.

Though most of our work from now on will be with tables, we'll only be concerned with *invariant* properties of our tables. This means we can't take the standard approach to relational composition we used in our example above. This doesn't work because, as we saw in the last section, the relation-numbers of the parts don't determine the relation-number of their cross product. We have no choice but to construct our compounds within some larger encompassing relation where its parts already exist. We can think of this larger relation as the *context* or *universe of discourse* of our construction.

Invariance means that our results must not depend on the *actual values* of this context relation, but only on which cell has the *same value* as which other cell. One way to insure this in practice is to work with structure tables, as defined in Chapter 2, with the understanding that the values 1, 2, 3 …n of the value index are to be regarded as abbreviations of *variables* v_1, v_2, v_3 …v_n. The criterion of a statement being invariant then becomes that we can replace it by an algebraic identity in the v_i. This trick is the key to replacing set theory by identity theory in our logical foundations, as mentioned in Chapter 2, section 1. But, as also mentioned, that belongs to another agenda.

Construction, as we ordinarily understand the word, implies interchangeable parts. If we *construct* C out of A and B, it is usually understood that we could just as well have constructed C out of A and B', where B' and B are the same *kind* of part. As we saw, being the same kind of relational part means being *congruent*. But being congruent means having the same *count table*. Thus the *counts* in our count tables belong to the very idea of composition. Counts are of course "natural numbers" in the old-fashioned sense, but they are also *natural numbers* in a new sense that is of a piece with Russell's idea of *relation-number*. As mentioned above, it's the ratios of these natural numbers that become probabilities and amplitudes in the core laws of quantum mechanics. We'll soon see why this is so.

To analyze composition structurally means that we can no longer think in terms of just adding parts and taking them away. Rather, we must now think in terms of *selecting* and *combining* the parts of a specified context. The count table resulting from a sequence of such selections and deselections is what we usually call a structure. However, the table itself can't be the whole story, since, *as such* it may contain little or no evidence of the parts that went into its construction. Thus it's not the table that should be called a structure, but the table plus something else having to do with the sequence of constructive acts which produced it. In order to put our finger on this something else, we'll introduce a new technical concept called *selection*, which will have roughly the same meaning here that it has in computer science.

3.3) Kinematic sequences

Selection, in brief, is a very abstract idealization of *presence*.

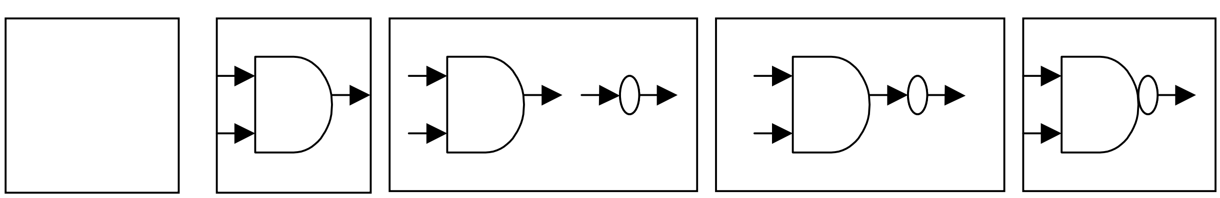
The word "present", like "past" and "future", is singular – there is only *one* present. *The* present, however, is a *changing* present. In our attempt to dispel or hide the baffling mystery of "changing sameness", we have invented the concept of *moments* which occur in *succession*. The concept of moments seems paradoxical, though – how can one thing, the present, be many? The pre-Socratics divided on this question by choosing one or the other horn of the dilemma; for Parmenides, the changing present is unreal, whereas for Heraclitus, it is the only reality. Modern science tries to embrace both. We'll try here to do something a bit different, which is to transform these two horns of the dilemma into two feet, so-to-speak, one planted in statics, the "other on the march" in dynamics. It's a neat trick if we can manage it.

When we are contemplating the world in the static mode, a presence is something *done*, a fait accompli, a thing with a *place* in history. To have a place in history means to have a potential place in a history book. But what is a history book? To read a history book and understand it *as such* is only possible if we switch to the kinematic mode and allow its printed words to conjure up a succession of presences that literally succeed each other in our imagination. But by so doing, we actually do penetrate the mystery of changing sameness – not by analyzing the changing present into something else, but by directly experiencing it. Moving on to the dynamic mode means that we also become attentive to *why* these "moments" succeed each other as they do, which is the same thing as being attentive to *what could have been done* to bring about a different succession.

A *kinematic sequence* will be defined as a sequence of *selected sub-tables* in a given table. Formalized in the language of set theory, this sequence is of course a perfectly static object. That doesn't mean we can assimilate it into pure mathematics, however – such an assimilation would amputate our marching foot. It's essential to keep alive the possibility of experiencing each step as *the* present. In this respect, a kinematic sequence resembles a phonograph record, or a movie, or a written play. On the one hand, such things are finished products, each having shape and structure we can study at leisure. On the other hand, they are designed to be *played back*. In this respect, a kinematic sequence is most like a play, which can be played back in various ways, with different scenery, costumes, timing etc., and is also a somewhat different play from the differing viewpoints of the author, the actors, and the audience. How a play is seen by the audience is *kinematics*, h how a play is produced is *dynamics*. The stage at each moment is the *selection*.

3.4) The NAND Gate

To illustrate this analogy, here is a very simple play in four acts called "The NAND Gate." It's plot goes like this: First AND comes onstage. Next, NOT comes onstage. Next, AND and NOT link up. Finally, their linked variables disappear, leaving the stage occupied only by the NAND gate, which is the happy marriage of the two characters who
have now fused into one. Fig 3.2 is a diagram of these four acts:

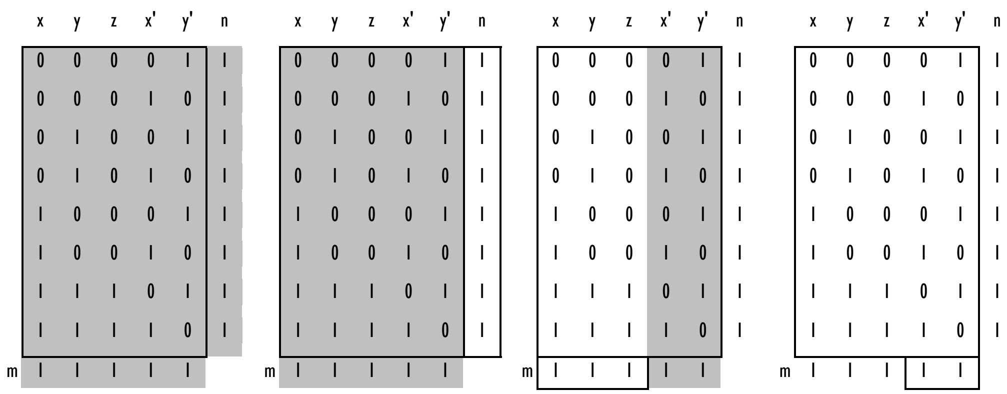


Curtain Act 1 Act 2 Act 3 Act 4

Fig 3.2 The NAND Gate (To see this figure in Word, go to Page Layout View)

Let's now translate this diagram into the language of tables and selections. Our tables will be count tables in which each row and column has a count of 1. Referring to Fig. 3, the foreground counts are shown in white, the background counts in gray; similarly the selected cells are white and the unselected cells are gray.

We start with a *context table* in which the AND and NOT gates are independent, and nothing is selected. This is shown as a *count table*, with n being the row count, m the column count. The first thing that happens is that all rows are selected. Since no columns have been selected yet, the cells all remain in darkness – this is only the curtain-raiser. Act 1 proper begins when AND comes on stage, which is accomplished by selecting its three columns. Act two brings on NOT by adding its two columns to the selection. Selected cells are shown in white, unselected cells gray.



Proscenium Lights dim columns — Opening curtain Select all rows — Act 1 Select AND columns — Act 2 Add NOT

Fig 3.3 Acts 1 and 2 of The NAND Gate

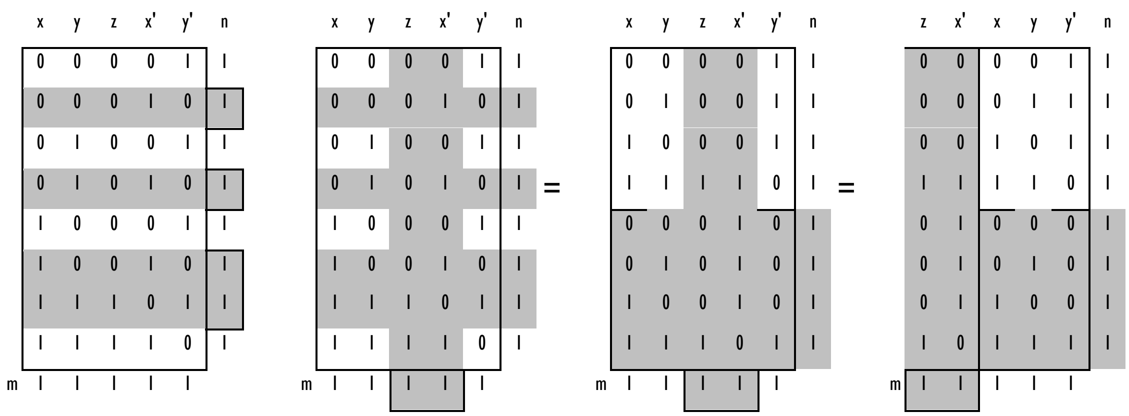
If all this seems a bit mysterious, it's because we haven't yet precisely defined all our terms. First of all, let's recall the definitions of part and sub-table in Chapter 2, section 2. A *part* is simply a set of rows or a set of columns. A *sub-table* is the spatial intersection of a set of rows (horizontal part) and a set of columns (vertical part), i.e. a sub-table is the set of all cells that are in both some row of a certain horizontal part and some column of a certain vertical part.

Selection: The selected sub-table. A kinematic sequence is a sequence of selected sub-tables, any sub-tables. A row is called *selected* if it belongs to the row-set of the selection, while a column is called *selected* if it belongs to the column-set of the selection. The set of selected rows and columns is called the *foreground*. A cell that is belongs to the *selection*, and will be called simply *selected*. Notice that no cell is selected unless the foreground contains both rows and columns.

Foreground: The set of all selected rows and columns. Notice that this is different from the *selection*, which is the *sub-table* (see above) defined by the rows and columns in the foreground.

Background: The complement of the foreground, i.e. the set of all unselected rows and columns. Since we usually start with a null selection, we'll use the term *background table* for the entire table with the selection ignored.

At the end of the second act, the two gates are independent, as we can see from the fact that every row of AND and NOT occur together in some row of the whole. Act 3 puts an end to this independence by *linking* the output of AND to the input of NOT, thus sending all rows in which the two disagree into the background (the second table of Act 3 is a permutation of the first that shows the selection as a block, and similarly the second table of Act 4.). In the final act, after harmony prevails, the harmoniously linked variables are themselves dispatched to the background, leaving selected only the table of a single gate, the NAND gate.



Act 3 Act 4 Regroup
Regroup
Link z and x' Hide connection Selected rows Selected
columns

Fig 3.4 Acts 3 and 4 of The NAND Gate

3.5) Dynamic sequences

A kinematic sequence is a sequence of frames which are snapshots of what is present at each successive moment. It contains no information, however, about how or why each frame changes into the next frame. When such information is supplied, we have a sequence of *dynamic frames*. A dynamic frame is a kinematic frame with an additional piece of information called an *action* which specifies how that frame is derived from the one before it. In this paper we'll only be concerned with actions that transfer rows and columns to and from the foreground.

Actions: For any given vertical part (column-set), there are two possible actions, the *V+ action*, which selects the unselected columns of that part, and the *V- action*, which deselects the selected columns of that part. *H+* and *H- actions* do the same for horizontal parts (row-sets). The part in question is called the *component* of the action.

Dynamic sequence: An initial selection together with a sequence of actions. Usually the initial selection will be null.

Let's now replay our four-act play as a dynamic sequence. Again, let's look at fig. 3.

We begin with an empty stage – the foreground is null. The first action, raising the curtain, is an H+ action which moves the eight rows of the table from the background to the foreground. These eight rows constitute Component 1, shown as a white box in the row-count column. There are still no selected columns, however, so the selection, i.e. the selected sub-table, is still empty (remember, the selection is defined as the set of all cells that are in *both* a selected row *and* a selected column.)

The second action, which occurs in Act 1, moves the three columns of the AND gate to the foreground (they must of course already exist in the background). This is a V+ action, which means that the set of these columns is a V+ component. Since all rows have already been selected, the cells whose rows and columns are now both selected make up a 3 by 8 sub-table, which constitutes the AND gate proper.

The third action, which occurs in Act 2, moves the NOT gate to the foreground. Its component contains the columns of NOT. The rows of NOT are already there, since they were added by the curtain raiser..

The fourth action (Act 3) correlates the output of AND with the input of NOT. This is done with an H- action which transfers all those rows to the background in which the NOT input differs from the AND output. Finally, in Act 4 the linked columns are moved to the background by a V- action whose component consists of the output column of AND and the input column of NOT.

3.6) Commuting dynamic sequences and empty parts

Dynamic sequences were designed to bring us more understanding of what we mean by structure. Are we then looking for a formal definition analogous to the one we found for shape? No, since that would be reducing structure to statics, which we saw is impossible. Is a structure then simply the history of a construction? No, history usually has too many *accidents*. Among the accidents of a history can be the order in which certain actions are performed. Act 2 of our play could just as well have occurred first, and Act 4 before Act 3. However, notice that Acts 1 and 2 have to come before acts 3 and 4, since you can't take something away before it has arrived. It would be nice to have to deal only with actions that *commute*, which would bring us closer to our intuitive notion of a structure as a shape that is intelligible as a timeless *assembly* of parts.

The need for order in a dynamic sequence comes from the fact that you can't move something from A to B before it has arrived at A. This is hardly a new problem in human affairs. Farmer A comes along with an ox and offers to sell it to farmer B for five pieces of silver. Farmer B will be getting five pieces of silver next week, but the ox is at hand now, and by next week it will have been sold to someone else. What is to be done? The solution is called *credit*.

For credit to become a system, two mathematical concepts are necessary, *zero* and *negative numbers*. These were unknown in the ancient world. When one thinks of the brilliance and sophistication of the best mathematics of that age, this is a surprising oversight. Fortunately it was remedied by the Arabs in the middle ages, which quickly led to double entry bookkeeping. What we'll now do is to adapt double entry bookkeeping to count tables, thereby making our actions commutative.

Roughly speaking, the trick is to expand the notation of count tables by allowing the arbitrary insertion of *empty* parts, i.e. parts whose count is zero. Such a part may contain rows or columns with non-zero counts, but each of these is balanced by a duplicate whose count has the opposite sign. The information conveyed by an empty part is that none its rows or columns exist in the table. Since this information is already conveyed by the absence of these rows and columns in the table proper, it is redundant, which means that adding an empty part to a count table has no effect on its *reference*. This is important to keep in mind: A count table with empty parts abbreviates the same relation table as that count table without its empty parts.

Empty row or column: Let C be a count table that abbreviates the table T. An *empty row* or *empty column* of C is a row or column whose count is 0. It supplies the (redundant) information that this row or column does not occur in T.

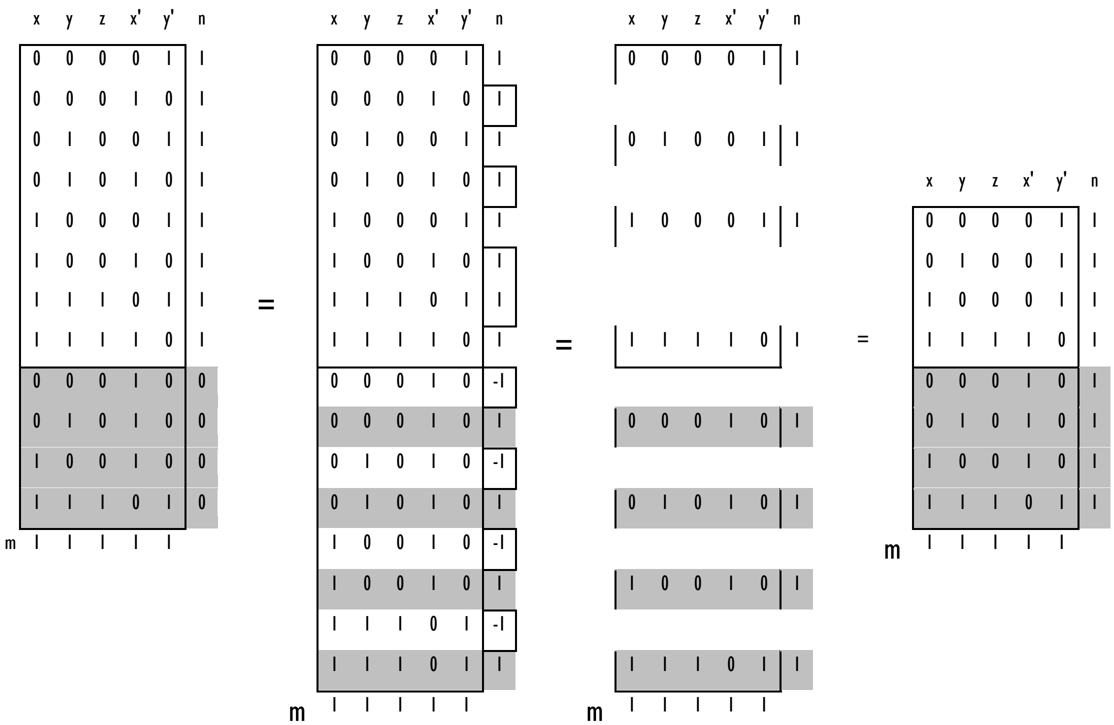
Empty pair: An *empty row pair* or *empty column pair* is defined as a pair of identical rows or columns, one of which has count 1 and the other count -1. An empty pair is to be regarded as synonymous with a single empty row or empty column, i.e. it signifies that this row or column does not occur in T.

Empty part: A part (row-set or column-set) whose members are either empty or belong to an empty pair in that set.

Empty table: A count table whose row set and column set are empty parts. An empty table abbreviates the *null* table, which is the table without rows or columns. To put it another way, it designates the *null* extension, i.e. *nothing*.

Selection was defined above for tables, but there is nothing to stop us from extending selection to count tables. Notice that if the positive half of an empty part has been selected, an action that *selects* the negative half of that empty part is equivalent to an action that *deselects* the positive half. The positive selection needn't occur first, however. If it does, the transfers are "proper", i.e. the positive part is present before it is taken away. If the negative selection occurs first, the foreground *goes into debt*, so-to-speak, in order to give the background a duplicate positive part, which the background may eventually return, thereby canceling the debt. Acts of selection, whether negative or positive, always commute, since each is simply a union of some component with the foreground, and the set operation of union is commutative. Thus if we replace all deselections in a dynamic sequence by negative selections, the result at the end is the same, but the dynamic sequence becomes commutative.

Let's replay The NAND Gate using these new commutative actions. Acts 1 and 2 are unchanged, since they only involve positive actions (see Fig 3.3). In Acts 3 and 4, however, the actions are negative. Above we showed them as deselections, but now we'll show them as *selections* of negative parts. Here (Fig 3.5) are the tables for Act 3, in which the AND gate is connected to the NOT gate by linking columns z and x'.



1) End of Act 2 2) Select our "debt" 3) Erase empty row part

Fig 3.5 Act 3 Linking z and x'

Act 3 links the columns z and x' by deselecting all rows in which their values differ. This is done as follows. First we add fours unselected *empty* rows to the count table which are duplicates of the four rows to be deselected; these are shown as the bottom four rows in 1) of Fig 3.5. Where do these empty rows come from? Since, according to set theory, the null set belongs to every set, they are already *there* in our table: we're merely acknowledging their presence when we add them to our count table abbreviation. The action of Act 3 consists of selecting the "negative half" of each of these four empty rows, thereby transferring the eight-row empty part to the foreground, and leaving the unselected positive duplicates in the background (Fig 3.5, 2). Erasing this empty part and compacting the resulting table leaves us with the AND gate selected, as in Fig. 3.4.

There is one wobbly step in this argument. Our empty-pair notational convention says that a negative row cancels a duplicate positive row. But suppose there are two such positive duplicate row? Which one gets cancelled? This makes no difference if nothing (or everything) has been selected, since duplicate rows are then in every way indistinguishable. But since, in our example, one duplicate is selected and the other is not, our dynamic sequence may have a very different outcome depending on which one is cancelled. There's an obviously right choice, but the rule for making it needs to be made explicit:

Empty pairing convention. A negative row (or column) cancels a duplicate positive row (or column) if and only if they are either both selected or both unselected.

Proper count table: A table in which for every selected negative row there is a duplicate selected positive row, and for every unselected negative row there is a duplicate unselected positive row. To put it another way, a table is proper if every negative part belongs to an empty part, and every empty part is either completely selected or completely unselected.

We'll allow improper count tables as intermediate steps in our dynamic sequences, provided such dynamic sequences always end up properly. Such dynamic sequences will exhibit "improper" structure. But does an improper table have an improper shape? Does it have any shape at all? Let's wait on that one.

Act 4 of our play is deselecting the linked columns z and x'. The action is essentially the same as in Act 3, but with columns instead of rows (See fig 3.6). First we show two empty columns that are duplicates z and x'. Then we rewrite the empty columns as empty pairs and select their negative halves, thereby incurring a "debt" to the background By the empty pairing convention, this moves the empty part into the selection; when we erase it, the foreground is left with the NAND gate, having "donated" its x and z' columns to the background.

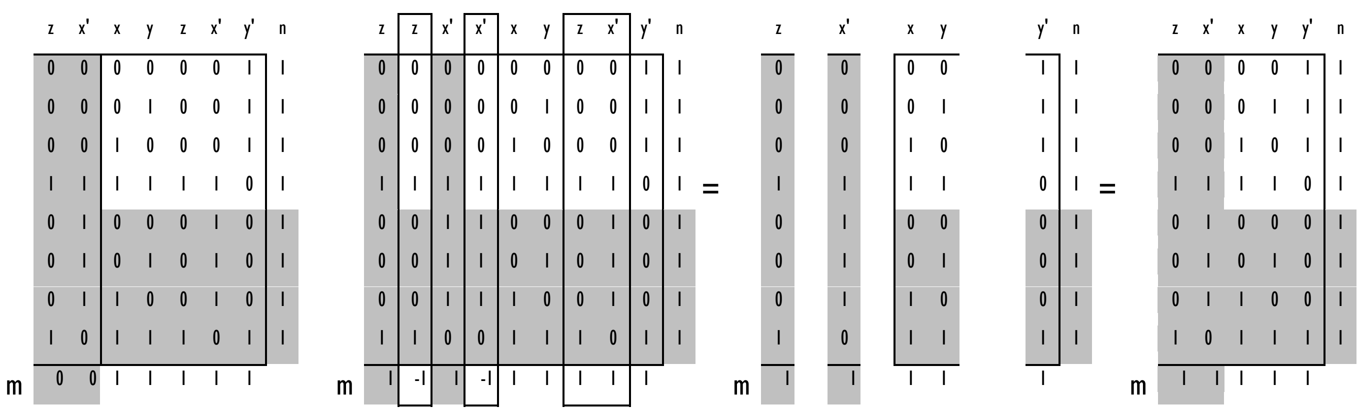


Fig 3.6 Act 4: Deselecting the linked columns z and x'.

3.7) The NAND Gate performed backward

Our motive for introducing zero and negative counts was to make our actions commutative. Let's now verify that we have actually done so by performing The NAND Gate backward (see Figs. 3.7, 3.8 and 3.9) Notice that all of the tables in the kinematic sequence are improper, until we finally come to the curtain raising, which of course occurs last, and which finally produces the proper table of the NAND gate.

Our backward sequence is not a reversal of our kinematic sequence, since that would start with the NAND gate and end up with nothing. Rather, the idea is to start with nothing selected and then perform the four actions of our play in reverse order to make sure we end up with the NAND gate selected.

Fig. 3.7 shows act 4, which is to select the negative half of a null z column and the negative half of a null x' column. The foreground is now in the hole by two columns; it has donated these to the background without yet having any compensating assets. The resulting table is of course improper.

	x	y	Z	x'	y'	n
	0	0	0	0	1	1
	0	0	0	1	0	1
	0	1	0	0	1	1
	0	1	0	1	0	1
	1	0	0	0	1	1
	1	0	0	1	0	1
	1	1	1	0	1	1
	1	1	1	1	0	1
m	1	1	1	1	1	

Context table

	z	z	x'	x'	x	y	z	x'	y'	n
	0	0	0	0	0	0	0	0	1	1
	0	0	1	1	0	0	0	1	0	1
	0	0	0	0	0	1	0	0	1	1
	0	0	1	1	0	1	0	1	0	1
	0	0	0	0	1	0	0	0	1	1
	0	0	1	1	1	0	0	1	0	1
	1	1	0	0	1	1	1	0	1	1
	1	1	1	1	1	1	1	1	0	1
m	1	-1	1	-1	1	1	1	1	1	

Incur column debt

Fig 3.7 Backward performance, Act 4

Fig. 3.8 shows the other three acts. Act 3 puts the foreground further in the hole by incurring a debt for the four rows in which z and x' disagree. Act 2, by selecting the NOT columns, pays the x' column debt, since the empty pair of x' columns are now in the selection. Act 1 pays the z column debt, so the table is now "vertically proper."

	z	x'	z	x'	x	y	z	x'	y'	n
	0	0	0	0	0	0	0	0	1	1
	0	1	0	1	0	0	0	1	0	1
	0	0	0	0	0	1	0	0	1	1
	0	1	0	1	0	1	0	1	0	1
	0	0	0	0	1	0	0	0	1	1
	0	1	0	1	1	0	0	1	0	1
	1	0	1	0	1	1	1	0	1	1
	1	1	1	1	1	1	1	1	0	1
	0	1	0	1	0	0	0	1	0	-1
	0	1	0	1	0	1	0	1	0	-1
	0	1	0	1	1	0	0	1	0	-1
	1	0	1	0	1	1	1	0	1	-1
	0	1	0	1	0	0	0	1	0	1
	0	1	0	1	0	1	0	1	0	1
	0	1	0	1	1	0	0	1	0	1
	1	0	1	0	1	1	1	0	1	1
m	1	1	-1	-1	1	1	1	1	1	

Act 3: Incur row debt

	z	x'	z	x'	x	y	z	x'	y'	n
	0	0	0	0	0	0	0	0	1	1
	0	1	0	1	0	0	0	1	0	1
	0	0	0	0	0	1	0	0	1	1
	0	1	0	1	0	1	0	1	0	1
	0	0	0	0	1	0	0	0	1	1
	0	1	0	1	1	0	0	1	0	1
	1	0	1	0	1	1	1	0	1	1
	1	1	1	1	1	1	1	1	0	1
	0	1	0	1	0	0	0	1	0	-1
	0	1	0	1	0	1	0	1	0	-1
	0	1	0	1	1	0	0	1	0	-1
	1	0	1	0	1	1	1	0	1	-1
	0	1	0	1	0	0	0	1	0	1
	0	1	0	1	0	1	0	1	0	1
	0	1	0	1	1	0	0	1	0	1
	1	0	1	0	1	1	1	0	1	1
m	1	1	-1	-1	1	1	1	1	1	

Act 2: Select NOT

z	x'	z	x'	x	y	z	x'	y'	n
0	0	0	0	0	0	0	0	1	1
0	1	0	1	0	0	0	1	0	1
0	0	0	0	0	1	0	0	1	1
0	1	0	1	0	1	0	1	0	1
0	0	0	0	1	0	0	0	1	1
0	1	0	1	1	0	0	1	0	1
1	0	1	0	1	1	1	0	1	1
1	1	1	1	1	1	1	1	0	1
0	1	0	1	0	0	0	1	0	-1
0	1	0	1	0	1	0	1	0	-1
0	1	0	1	1	0	0	1	0	-1
1	0	1	0	1	1	1	0	1	-1
0	1	0	1	0	0	0	1	0	1
0	1	0	1	0	1	0	1	0	1
0	1	0	1	1	0	0	1	0	1
1	0	1	0	1	1	1	0	1	1
1	1	-1	-1	1	1	1	1	1	

Act 1. Select AND

Fig 3.8 Backward performance, Acts 3, 2 and 1

We have now come to the beginning of Act 1, but notice that our selection still bears no resemblance to the NAND gate – indeed, it's not even proper. What went wrong? Actually, nothing. Don't forget the curtain raiser! Remember, this was the first action of our forward dynamic sequence, and it involved selecting all of the rows in the context relation table. In the present performance, this last action gives the foreground the assets to pay back its row debt, so all the negatives are now properly cancelled. The resulting proper table does indeed turn out to be our NAND gate (Fig 3.9):

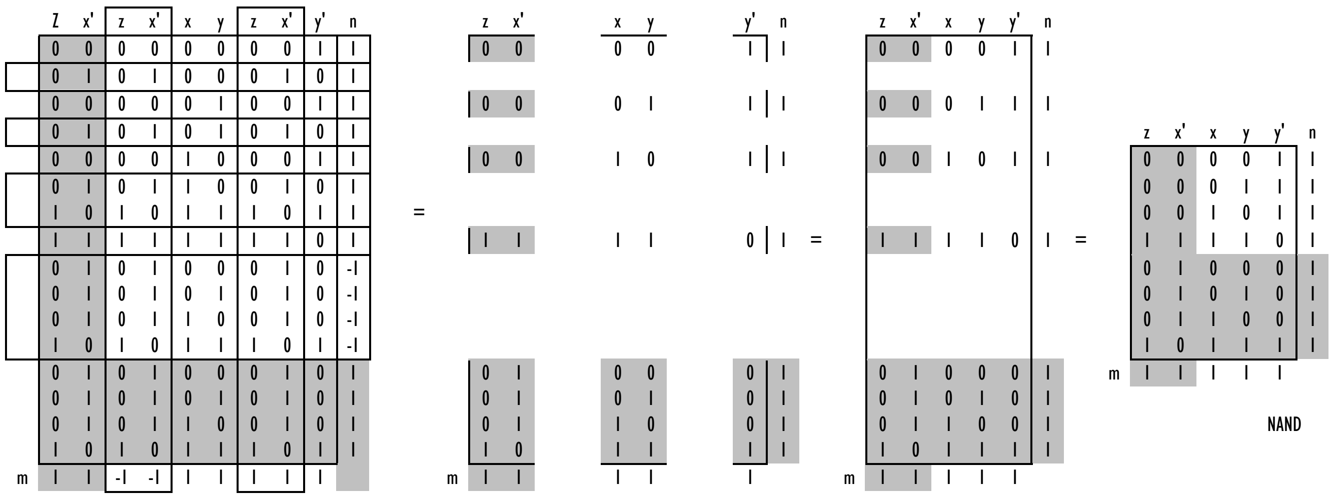


Fig 3.9 Backward performance: Opening curtain: Select all relation table rows and erase the resulting empty parts

3.8) Timeless change

…Commutative dynamic sequence as a *timeless* set of actions etc. (very brief)

CHAPTER 4. Past, Future and Quantum

We went to a fair amount of trouble in the last chapter to develop a strange new notation for moving rows and columns in and out of the selection. Our nominal goal was to make the actions of a dynamic sequence commute, which indeed they now do. This new notation also simplifies our formalism by getting rid of the need for actions that deselect. However, so long as a dynamic sequence of the new type begins and ends with proper tables, we can always find a sequence of the old type which does the same job, so one might question whether these formal gains are worth all the trouble.

As the reader may have suspected, there is something else at stake here besides formal simplicity, and that something else has to do with quantum mechanics. The present chapter will provide an informal survey of the emergence of quantum mechanics from structure theory. We'll see in broad outline how the formalism of commutative actions explains the cancellation of probability amplitudes, the square law for probabilities, and the generalized von Neumann dynamical law. In order to do so, however, we must expand that formalism by bringing in the concepts of past and future. The key move here is to separate *past-future polarity* from the concept of *temporal succession*, just as we separated change from sequence in the last chapter.

The mathematical details of this derivation will be postponed until the next two chapters so that we can concentrate here on the basic ideas.

4.1) Past and future in physics

There is a long-standing scandal in physics which, despite an occasional exposé, has been pretty well kept in the closet. It has to do with the physical difference between past and future. The scandal is that there *doesn't seem to be any*. The basic laws of both Newtonian mechanics and quantum mechanics are completely invariant under time-reversal.

In section 3.3, *selection* was defined as an idealization of *presence*, and we have managed so far to discuss kinematics and dynamics almost entirely in the present tense. We did discretely mention the *beginning* and *end* of our sequences, but *past* and *future*, those mysterious inseparable companions of the present, have not yet shown their faces. Scandalous or not, it's time for them to do so.

The past is what we remember and read about, the future what we plan and work for. The past is what is over and done with, the future what we choose it to be. The past is what we regret or take pride in, the future what we fear or anticipate or hope for. How could any two things be more utterly different than past and future? And yet physics says they are the same!

It's our habit, when confronted with a puzzle, to ask for a *solution*, as when we ask for the solution to a set of equations. This is not something we can expect in the present case. The mystery of past-future symmetry will forever remain *unsolved*. But *solving* a mystery is not always the same thing as *penetrating* it, which can also result from our being informed by new kinds of *experience*.

Past-future symmetry takes very different forms in statics, kinematics and dynamics.

In statics it presents no mystery at all. If one unrolls the movie film across the cutting table, past is left and future is right; to reverse past and future, just go to the other side of the table. Or, to switch the metaphor, the road stretches from East to West across the hills by the shortest path (Maupertuis' principle of least action), and that same shortest path leads from West to East.

In kinematics, past-future symmetry is more of a problem, since it would require us to regard a movie run backwards as a realistic story, even though we saw old people growing young, manuscripts emerging from the roaring flames, etc. The course of events in such a reversed showing would in any case still be recognizable *as* a course of events, however.

This is no longer true when we come to dynamics. How could we remember the future? It hasn't happened yet. How could we choose the past? It's already over and done with. How could causes come after their effects? When we start to ask not only what happens but *why*, we quickly discover that common sense is hopelessly unidirectional.

To fully assimilate the symmetry of past and future into our dynamical experience of the world will probably require a radical change in the nature of human experience itself. This is not a scientific project, or at least not a project for the science of today. But there are still things to be learned from today's science that point us in the right direction. Which brings us to quantum mechanics.

Our present task is to add *past* and *future* to the language of dynamic sequences in a way that makes their symmetry *intelligible*. We'll see that the generalized quantum core is an immediate consequence of expanding our language in this way. The quantum rules in particular hold for those dynamic sequences where past-future symmetry is not only intelligible but *true*. To put it more starkly, *past-future symmetry alone is what defines the quantum domain.*

Our point of departure this time will not be computer science, which gave us *selection*, but the words of Saint Augustine, who will give us another puzzle.

4.2) Augustine on time

"What, then, is time? If no one asks of me, I know; if I wish to explain to him who asks, I know not." is one of Augustine's better-known sayings [*Confessions*, Book XI, Chapter 14] Later he goes on to say something more challenging, which is that the past and future don't *exist*.

"What is now clear and unmistakable is that neither things past nor things future have any existence, and that it is inaccurate to say, 'There are three tenses or times: past, present and future,' though it might properly be said, 'There are three tenses or times: the present of past things, the present of present things, and the present of future things'. These are three realities in the mind, but nowhere else as far as I can see, for the present of past things is memory, the present of present things is sight, and the present of future things is expectation. If we are allowed to put it that way, I do see three tenses or times, and admit that they are three." [*Confessions*, Book XI, Chapter 20]

Bertrand Russell thought highly of this idea. "..It is clearly a very able theory, deserving to be seriously considered. I should go further, and say that it is a great advance on anything to be found on the subject in Greek philosophy. It contains a better and clearer statement than Kant's of the subjective theory of time – a theory which, since Kant, has been widely accepted among philosophers." [*A History of Western Philosophy*, p354] Russell himself does not go so far as to accept it, though. "I do not myself agree with this theory, in so far as it makes time something mental."

But does it make time something mental? Augustine's statement that "These are three realities in the mind, but nowhere else as far as I can see …" doesn't seem to leave much room for other interpretations. However, we can scarcely regard his few passing remarks on past and future as a finished theory, and there are other ways to carry his ball. For instance, instead of treating memory, sight and expectation as different *mental activities*, as Augustine does, we can treat them as *different ways of seeing things*. On this interpretation, it would not be my actual thoughts but the *presence* of my *present way of seeing* that brings something past or future into the present.

Let me give an analogy. I am facing the window and I see that the couch is on my right. I think "The couch is on my right." My thoughts then turn to other things. The way I am facing, though, remains the same, so *my way of seeing things as left or right* remains the same. The couch remains on my right, not because it's *in my mind* that it's on the right, but because it *is* on my right.

To carry the analogy through, my faculties of *memory*, *sight* and *expectation*, taken together, constitute a certain "temporal orientation" which determines how I see things as past, present or future. Of course it conditions how I think, but it does so as a condition of my objective situation, not of my mind. If Augustine's remarks are understood in this way, they suggest a new principle of *the relativity of past and future* which could take its place alongside the relativity of motion and simultaneity.

The identification of past and future with memory and expectation is of course an oversimplification. A full catalogue of our ways of temporal "seeing" would include many other modes: change, rhythm, repetition, speed, continuity, uncertainty, surprise, choice, hope, fear, regret etc., all of which involve past and future in ways that are not reducible to memory, sight and expectation. But these three modes have a kind of definitive simplicity: Memory is facing the past, sight is facing the here and now, and expectation is facing the future. Of all modes, these most directly capture the three *tenses*. Furthermore, taken together they again reveal the paradoxical nature of time, though now seen as a paradox of parts and wholes rather than of changing sameness. Sight, or seeing, is one possible mode out of three. But it also encompasses the other two: The word "was" means "*is* past," and "will be" means "*is* to come."

Discussions of past and future in physics almost always start out with a temporal series in which they are defined as earlier and later in the sequential order. The significance of Augustine's account of past and future for our present work is that it shows us that we don't need such a sequence. He forcefully reminds us that that we have a direct *present* awareness of past and future that stands alone as such. In modern terms, his is a *local* rather than a *global* definition. Notice that this makes the concepts of past and future available to theories of time that allow other than linear temporal ordering, for instance time loops, or even multiple dimension of time. More important here, it makes it available to our abstract dynamics of structure, as we shall soon see.

When past, present and future are seen together, the present presents itself as the *link* between the other two. What, then, is a link? In contrast to a bare identity, a link is something *removable*, and indeed we do sometimes remove the link of the present in imagination, as when we contemplate the future starting *now,* never mind the past, or the past ending *now*, never mind the future. Thus our experience of time countenances three *now*'s: the end of the past, the beginning of the future, and these two understood as one and the same. When we are in the mode of asking how and why, all three of these *now*'s belong to our concept of *state*.

Structure theory is not phenomenology, and can make no pretence of becoming a descriptive framework for such experiential modes. It can, however, offer a candidate for the *structure* of the present *as* a link. We've already casually met this candidate several times in Chapter 3 as that horizontal component called a *link* which deselects the rows in which two column tables disagree. Actually, it's not just this component alone that has the structure of the present as link, but it together with the two column tables that it links. These three, when they are both given as components and selected together, are what structure theory offers as its abstract precursors to the end of the past alone, the beginning of the future alone, and that which makes them the same.

4.3) Actual, possible, impossible and potential (incomplete)

Past and future in daily life are closely tied to *actual* and *potential*. To *act* is to *actualize* some part of what is potential, leaving the remainder unrealized, and perhaps unrealizable. Those potentialities which, in retrospect, have become unrealizable, become *alternatives* to the actual. When we speak of alternatives for the future, we are looking forward to a fight to the death among potentialities. Action is not always so bellicose, however. Sometimes it follows a blind path, oblivious to the alternatives, and by so doing begets new and unimagined potentialities. Deliberative action, which is that mode of action that moves towards a predefined goal, always does so at the expense of the potential. Deliberative action creates the common-sense contrast between the future as *open* and the past as *closed*, with the present as the occasion of *closure*.

The ideal of deliberative action sets the stage for technology, and for those sciences that are valued for furthering technology. For most of us today, this stage is our world, or at any rate, our waking world. To see beyond it requires first of all that we observe it in a way that goes beyond its entertainment value to its essential features. Here, in brief, is what defines "techno-world":

The past is entirely actual, or more exactly, what is past and done with is entirely actual. In brief, the past is a history book, entirely present, if not to us then to an imagined super-mind whose awareness is what we call *fact*. We sometimes speak of "what would have been possible if …," but this is an act of imagination in which we place ourselves at an earlier time looking toward the future. The future itself is open to many possibilities. We experience this most directly when we are undecided about how or whether to act. If Augustine had lived in our more optimistic times, he might have written "..the present of future things is expectation and opportunity." We also speak of "the possibilities" when we don't know what will happen. When we do know what will happen, or think we know, the future becomes "expectation"; though it is not *seen* as actual, it is *imagined* as actual, i.e., as "..the present of future things."

….. to be continued.