An Analysis of an Epistemic Toy Theory of Quantum Mechanics with Connections to the Stabilizer Formalism and Local Hidden Variable Tables.

Kyle W. Martin

Advisor: Carlton M. Caves

2009-2010

Toy models are used in every field of physics to help physicists understand more complex systems without having to deal with the entire theory. Pictures also help us understand systems with greater efficiency. For instance, we use Feynman diagrams to deal with scattering cross-sections. Robert Spekkens developed a toy theory of quantum mechanics and published his final version in Physical Review A in 2007. His toy theory is based on a simple principle but recreates a great deal of phenomena thought to be purely quantum in nature. His theory deals with the states of a spin-$1/2$ *particle, or qubit, and, specifically, with the qubit states that lie on the Cartesian axes of the Bloch sphere. He uses pictures to describe these states and how they are manipulated. We show that this toy theory is closely related to the fully quantum-mechanical stabilizer formalism, which deals with the same qubit states. We also show that the transformations that Spekkens develops are just normalizer elements, transformations that take stabilizer states to stabilizer states. We also define these normalizer transformations on the generator matrices that describe the connections between the stabilizers and the local hidden variable tables and use these generator matrices to connect this toy theory to the stabilizer formalism.*

Acknowledgments

I would like to thank Carl Caves for his help and guidance during this undergraduate thesis project. I would also like to thank Matt Elliott for his help in understanding the connections between the stabilizer formalism to the local hidden variable tables that can be found in his thesis. I would like to thank the endless list of people that I had conversations with and all those who helped my edit me final paper.

Contents

I. TOY THEORY

Robert Spekkens developed his toy theory over a long period, posting the first version to the e-print arXiv in January 2004 and publishing the final version in Physical Review A in 2007. His toy theory is not based in quantum mechanics, and is not equivalent, but it recreates some phenomena often thought to be purely quantum in nature. The toy theory can be used to investigate such phenomena as superdense coding, teleportation, and no-cloning. We examine the derivation of Spekkens's toy theory in order to develop connections to other models of quantum behavior. Spekkens limits the scope of his original toy theory to a case of three systems. We have formulated his toy theory in a way to expand beyond the initial scope and apply the toy theory to a N-system model.

Spekkens introduced his toy theory by carefully distinguishing the ontic and epistemic states of a physical system. An *ontic state* (derived from the Greek *ontos*, meaning "to be") is a state of reality. An *epistemic state* (derived from the Greek word *episteme*, meaning "knowledge") shall be what we refer to as a state of knowledge. The easiest way to distinguish ontic states and epistemic states is to refer to physical examples. Ontic states are usually seen in classical mechanics where we can determine all the parameters that define a system, such as position, momentum, and energy. Epistemic states are usually introduced in statistical mechanics when the value of a parameter is difficult, if not impossible to determine. An example of an epistemic state in statistical mechanics would be the probability distributions used for the positions, momenta, and energies of gas molecules in a room. Spekkens created a toy model for two-level quantum systems, or *qubits*, which defined the possible epistemic states as obeying a well-defined principle, spelled out just below, which restricts knowledge of the ontic states. He found that upon doing so the resulting theory he developed had some effects thought to be purely quantum in nature. Of course, since Spekkens's ontic states

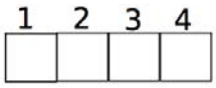

FIG. 1: The box representation for an arbitrary elementary state, labeling the ontic states from left to right

are like local hidden variables, it is known that his toy model cannot account for all the predictions of quantum mechanics for multiple qubits. [10]

A. Elementary systems

1. Ontic and epistemic states and the knowledge balance principle

Spekkens's toy theory is based on the following *knowledge balance principle*:

> *If one has maximal knowledge, then for every system, at every time, the amount of knowledge one possesses about the ontic state of the system at that time must equal the amount of knowledge one lacks.*

In other words, we can obtain at most half the information about the true ontic state of a system. In order to make use of this principle, there must be a concrete specification of the ontic states and a representation of knowledge about the ontic states. For example, consider a system with four ontic states, labeled 1, 2, 3, and 4. Figure 1 shows this example. We could identify a particular ontic state by asking a series of four canonical yes/no questions, "Is it 1, or not?", "Is it 2, or not?", etc., but this is an inefficient questioning scheme. We could alternatively ask just two questions to identify a particular ontic state of this system: "Is it in the set {1,2}, or not?", and "Is it in the set {1,3}, or not?". A system of four ontic states requires the answer to exactly two such canonical yes/no questions to determine a particular ontic state. An epistemic state of maximal knowledge satisfying the knowledge balance principle corresponds to knowing the answer to exactly one of the canonical yes/no questions.

We cannot apply Spekkens's knowledge balance principle to a system of two ontic states because a system of two ontic states requires the answer to just one canonical question and therefore cannot fulfill the knowledge balance principle. A system with four ontic states is

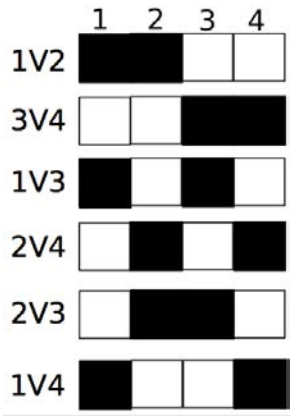

FIG. 2: Box representation of all of the epistemic states for an elementary system, and their state labels.

thus the smallest system that can fulfill the knowledge balance principle; we refer to a system of four ontic states as an *elementary system*. Throughout this thesis, we are only concerned with the epistemic states that correspond to maximal knowledge; for a single elementary state, these require the answer to exactly one canonical yes/no question. There being three unique canonical questions and two possible answers to each question, we conclude that there are six epistemic states of maximal knowledge in an elementary system.

If we ask the question "Is it in the set {1,2}, or not?" with result +1 (yes), we know that the state is 1 or 2. We define the symbol $\vee$, to be read "or," and this epistemic state is written $1 \vee 2$. The six possible epistemic states of maximal knowledge are the following:

$$1 \vee 2\,, \quad 3 \vee 4\,, \quad 1 \vee 3\,, \quad 2 \vee 4\,, \quad 2 \vee 3\,, \quad 1 \vee 4\,. \tag{1}$$

We can create a visual representation of the ontic states by labeling a series of four boxes, as in Figure 1. For an epistemic state, we shade the boxes that satisfy the conditions of the canonical question, as in Figure 2. [10]

When we answer one of these canonical questions we get a epistemic state that Spekkens later connects to quantum states. This canonical question can be thought of as a quantum operator. Although Spekkens never formally connects his canonical questions to operators in quantum mechanics mechanics, we would like to connect these questions to operators in quantum mechanics. Taking the idea that Hermitian operators are observables, we can think of a canonical question as a two-valued observable. A yes or no answer to the question "Is it in the set {1,2}, or not?" will be defined as the Z observable, whereas a yes or no

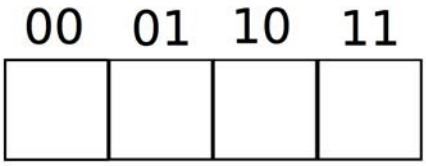

FIG. 3: The four ontic states of an elementary system labeled from left to right in binary. The first digit corresponds to the toybit b and the second digit corresponds to the toybit a.

answer to the question "Is it in the set {1,3}, or not?" will be called the X observable. The observables will have value $+1$ for a "yes" answer and -1 for a "no" answer. These two observables, under multiplication, generate the group that describes answers to all canonical yes/no questions. Because the multiplication of a generator with itself will always have a $+1$ result, we will define X^2 and Z^2 both to be the identity I, which always has a "yes" answer. The final group element, which we will call Y, will be defined to be the product $\pm XZ$, with the sign depending on how we ask the third canonical yes/no question. We will refer to this freedom in the sign of the XZ product as $W = XYZ = \pm 1$. Figure 2 shows the box representations we use for these states.

It is usually easier to work in a bit representation, so we now define the binary "toybits" a, b, and c seen in Equation 2

$$\begin{aligned} Z &= (-1)^b \,, \\ X &= (-1)^a \,, \\ W &= (-1)^c \,, \quad \text{which makes} \\ Y &= (-1)^{a+b+c} \end{aligned} \tag{2}$$

All addition of these binary toybits will be done mod 2, as is automatic in the exponent of -1. We can relabel the boxes in our visual representation with the binary digits a and b, as shown in Figure 3. After we have relabeled the boxes, our binary toybits b and a emerge as the labels of epistemic states. Simply declaring the value of one of these binary toybits results in four of the epistemic states for an elementary system, as shown in Figure 4.

Now let's consider the toybit c. The value of the observable $Y = XZW = (-1)^{a+b+c}$ is determined by $a+b+c$. The two choices for c don't change the final two epistemic states, but they do change the way these two elementary states correspond to a value for Y. Figure 5 shows the final two epistemic states if we were to choose $c = 0$. If we conversely chose $c = 1$,

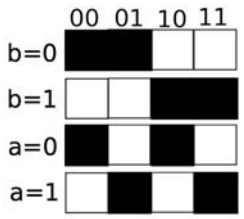

FIG. 4: The first four epistemic states, each depending only on the value of one of the a or b toybits.

the roles of the representations are reversed, as shown in Figure 6. Whenever we consider the value of Y for an epistemic state, we must define the value of c. To put it differently, to read off the value of Y from a box representation, we must have the value of c.

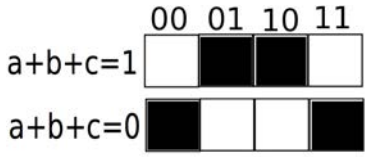

FIG. 5: The final two epistemic states, which are determined by $a + b + c$, here with the choice of $c = 0$.

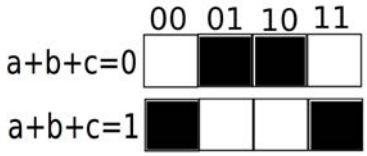

FIG. 6: The final two epistemic states, which are determined by $a + b + c$, here with the choice $c = 1$. These two epistemic states are the same as in Figure 5, but the way they correspond to a value of Y has been reversed.

We now have all of the necessary apparatus to describe the states of a states of a single system in Spekkens's toy theory. To fully evaluate his theory, however, we must also examine the allowed transformations between states and measurements on these states.

2. Measurements of an elementary system

Measurements in toy theory are the canonical questions themselves, measurements determine the toy bits defined in Equation 2. As soon as a measurement is performed on the elementary state, the state is projected into the epistemic state corresponding to the answer obtained. For instance, if we have an elementary system in the epistemic state $b = 0$ $(1 \vee 2)$ and we inquire about the value of X, we will project into the state $a = 1$ or $a = 0$ depending on the answer. To retain the knowledge balance principle, the system must forget its original ontic state before we inquired about X. In other words, although we know the system was in state $b = 0$ before the measurement, and we learn, say, that the system is in state $a = 0$ $(1 \vee 3)$, we cannot conclude that the system is in the ontic state 00. Instead, all we know after such a measurement is that the system is in one of the two ontic states corresponding to the measurement result $a = 0$ $(1 \vee 3)$. [10]

3. Transformations of an elementary system

Spekkens never makes the operator connection and never labels his boxes with binary numbers. When he defines the allowed transformations on an elementary system he continues to use a base ten labeling scheme. This labeling scheme is more efficient when defining the transformations on elementary systems so we will return to the base ten labeling scheme here. There are three types of transformations in an elementary system. The first is a many-to-one transformation. For example, we can consider the transformation that takes the epistemic state $1 \vee 2$ to the ontic state 1. This new state is no longer an epistemic state and is outside the scope of the theory, so we will neglect this type of transformation. The second type is a one-to-many transformation, an example of which would take $1 \vee 2$ to $1 \vee 2 \vee 3$; this state no longer has maximal knowledge as defined by our knowledge balance principle. Because we are only interested in states that have maximal knowledge, we will also neglect this type of transformation. The third and final type is a one-to-one transformation. This type can be thought of as a permutation of the four boxes. We can thus use permutations to characterize the transformation. For example, a transformation that takes $1 \rightarrow 1$, $2 \rightarrow 3$, $3 \rightarrow 4$, and $4 \rightarrow 2$ consists of a one-cycle and a three cycle permutation. We would write this transformation as $(1)(234)$, where numbers in parentheses are cycled in a right-handed

(1^4)	(31)	(21^2)	(2^2)	(4)
(1)(2)(3)(4)	(234)(1)	(12)(3)(4)	(12)(34)	(1234)
	(243)(1)			(1432)
		(13)(2)(4)	(13)(24)	
	(134)(2)			(1243)
	(143)(2)	(14)(2)(3)	(14)(23)	(1342)
	(124)(3)	(23)(1)(4)		(1324)
	(142)(3)			(1423)
		(24)(1)(3)		
	(123)(4)			
	(132)(4)	(34)(1)(2)		

FIG. 7: All of the allowed transformations on an elementary system grouped into cycle classes and paired with their inverses. Figure from [10]

fashion. Permutations with the same number of cycles form a class. There are 24 possible one-to-one transformations, grouped in five classes, as shown in Figure 7.

Permutations that are paired are inverses, and permutations that are not paired are their own inverse. These cyclic permutations are the only way in which we can move between states of elementary systems. From one epistemic state and all of the transformations, we can build any other epistemic state. We now turn to enlarging the theory to include more than one elementary system; the main addition is presence of epistemic states that codify correlations between elementary systems. [10]

B. Two elementary systems

1. Ontic and epistemic states

A combination of two elementary systems requires four toybits to fully specify a particular ontic state. A state of maximal knowledge, according to the knowledge balance principle, would be a state where exactly two toybits are known. The symbol "·" to be read "and" will help us to introduce the idea of a multiple system.

At the two-system level, there are two different types of epistemic states that satisfy the knowledge balance principle. In the first type, we can address each individual elementary system separately. We can write these epistemic states as separate states of the two elemen-

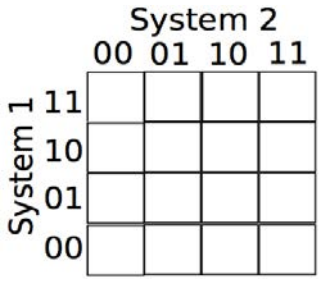

FIG. 8: Box representation for an arbitrary two-system state. The first system's toybits are labeled from bottom to top as the vertical boxes, while the second system's bits are labeled right to left on the horizontal boxes.

tary systems, with our new symbol connecting them. For instance, $(1 \vee 2) \cdot (1 \vee 2)$ would be the system with toybit constraints $b_1 = 0$ and $b_2 = 0$. In the second type of two-system epistemic state, we cannot separate the elementary systems involved, since the bits of information about the individual systems are correlated. Consider, for example, the system with toybit constraints $a_1 + a_2 = 0$ and $b_1 + b_2 = 0$; we can use our new symbol to write this state as $(1 \cdot 1) \vee (2 \cdot 2) \vee (3 \cdot 3) \vee (4 \cdot 4)$. We cannot clearly define a toybit for one system without defining the same toybit for the other system.

We can invent a picture to capture all of the two-system states, but we will have to extend our original picture into a second dimension. Figure 8 shows that the visualization requires four toybits to fully specify an ontic state. A state of maximal knowledge, according to the knowledge balance principle, would be a state where only two toy bits are known.

Not all possible ways of filling in four boxes are acceptable, however, since we must be careful not to violate the knowledge balance principle for either of the single elementary systems by itself. We can still only have at most one toybit of information about each individual elementary system. Figures 9 and 10 show the states described above, which do satisfy the knowledge balance principle for each separate system. All of the boxes that satisfy the toybit equations for these states are shaded. It turns out that permutations of the rows and columns of these pictures, which correspond to the single-system transformations already discussed, yield all of the allowed epistemic states for two elementary systems.

An example of a four-box state that violates the knowledge balance principle is $(1 \cdot 1) \vee (2 \cdot 1) \vee (3 \cdot 1) \vee (4 \cdot 1)$, which is specified by toybits $a_2 = 0$ and $b_2 = 0$ and corresponds to shading the first column of boxes. This state obviously violates the knowledge balance

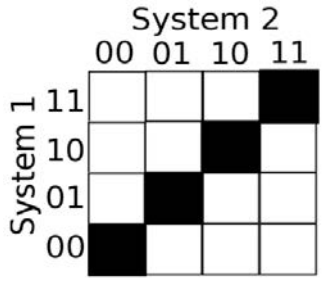

FIG. 9: The box depiction of the epistemic state $(1 \cdot 1) \vee (2 \cdot 2) \vee (3 \cdot 3) \vee (4 \cdot 4)$. All of the boxes are shaded such that the linear equations $a_1 + a_2 = 0$ ($a_1 = a - 2$) and $b_1 + b_2 = 0$ ($b_1 = b_2$) are simultaneously satisfied. The two elementary systems are correlated in this epistemic state.

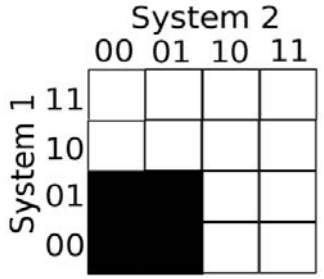

FIG. 10: The box depiction of the state $(1 \vee 2) \cdot (1 \vee 2)$. All of the boxes are shaded such that the linear equations $b_1 = 0$ and $b_2 = 0$ are simultaneously satisfied. In this situation, each elementary system has its own separate state.

principle because we know the ontic state of the second system (without knowing anything about the ontic state of the first system). Figure 11 shows another state that violates the knowledge balance principle in a slightly less obvious way. Even so, we can quickly see how this system violates the knowledge balance principle if we ask for the value of Z_1. Upon obtaining the answer to this question, the system would be projected into the $b_1 = 0$ or $b_1 = 1$ state; in both cases, we would then have complete knowledge of the ontic state of system 2 as either $a_2 = 0 = b_2$ for $b_1 = 0$ or $a_2 = 1$ and $b_2 = 0$ for $b_1 = 1$. [10]

It is worth spelling out in detail the allowed epistemic states of two elementary systems, and it is easiest to do so in terms of the toybits. For a single elementary system, we can constrain the value of a, b, or $a+b$. Formally, we need to consider the algebra of these toybits and, in addition, the no-constraint condition, which we label by a boldface $\mathbf{0}$. We let d be a

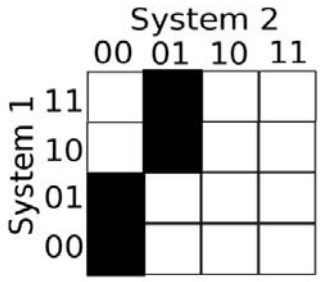

FIG. 11: The state depicted here, $(1 \cdot 1) \vee (2 \cdot 1) \vee (3 \cdot 2) \vee (4 \cdot 2)$, violates the knowledge balance principle. This state satisfies the simultaneous linear equations $b_1 + a_2 = 0$ ($b_1 = a_2$) and $b_2 = 0$. These linear equations make clear that if one determines the value of Z_1, i.e., determines b_1, then one knows the values of both a_2 and b_2, in violation of the knowledge balance principle.

variable that can stand for any of these four quantities. For two elementary systems, we need two linearly independent toybit constraints, on $d_1 + d_2$ and $d'_1 + d'_2$. Linear independence implies that both these quantities are not $\mathbf{0}$ and that they are not equal, which is equivalent to the following conditions:

$$\begin{aligned} &d_1 \neq \mathbf{0} \text{ or } d_2 \neq \mathbf{0} \,, \\ &d'_1 \neq \mathbf{0} \text{ or } d'_2 \neq \mathbf{0} \,, \\ &d_1 \neq d'_1 \text{ or } d_2 \neq d'_2 \,. \end{aligned} \tag{3}$$

Now let's consider the consequences of the knowledge balance principle, motivated by the discussion in Figure 11. If $d_1 = \mathbf{0}$ or $d'_1 = \mathbf{0}$ or $d_1 = d'_1$, then d_2 and d'_2 are already determined (if $d_1 = d'_1 = \mathbf{0}$), or observation of the nonzero one of d_1 and d'_1 determines both d_2 and d'_2 on the second system; if $\mathbf{0} \neq d_2 \neq d'_2 \neq \mathbf{0}$, this would determine the ontic state of the second system, in violation of the knowledge balance principle. Thus we conclude that if $d_1 = \mathbf{0}$ or $d'_1 = \mathbf{0}$ or $d_1 = d'_1$, the knowledge balance principle requires that $d_2 = \mathbf{0}$ or $d'_2 = \mathbf{0}$ or $d_2 = d'_2$. We can, of course, reach the same conclusion with the two systems reversed: if $d_2 = \mathbf{0}$ or $d'_2 = \mathbf{0}$ or $d_2 = d'_2$, the knowledge balance principle requires that $d_1 = \mathbf{0}$ or $d'_1 = \mathbf{0}$ or $d_1 = d'_1$.

Putting together the requirements of linear independence and the knowledge balance

principle, we find the following allowed possibilities:

$$\begin{aligned}
&\text{(i)} \quad d_1 = \mathbf{0},\ d_1' \neq \mathbf{0},\ d_2 \neq \mathbf{0},\ d_2' = \mathbf{0}\ , \\
&\text{(ii)} \quad d_1 \neq \mathbf{0},\ d_1' = \mathbf{0},\ d_2 = \mathbf{0},\ d_2' \neq \mathbf{0}\ , \\
&\text{(iii)} \quad d_1 = \mathbf{0},\ d_1' \neq \mathbf{0},\ d_2 = d_2' \neq \mathbf{0}\ , \\
&\text{(iv)} \quad d_1 \neq \mathbf{0},\ d_1' = \mathbf{0},\ d_2 = d_2' \neq \mathbf{0}\ , \\
&\text{(v)} \quad d_1 = d_1' \neq \mathbf{0},\ d_2 = \mathbf{0},\ d_2' \neq \mathbf{0}\ , \\
&\text{(vi)} \quad d_1 = d_1' \neq \mathbf{0},\ d_2 \neq \mathbf{0},\ d_2' = \mathbf{0}\ , \\
&\text{(vii)} \quad \mathbf{0} \neq d_1 \neq d_1' \neq \mathbf{0},\ \mathbf{0} \neq d_2 \neq d_2' \neq \mathbf{0}\ .
\end{aligned} \tag{4}$$

This looks like a mess, but a little thought shows it is not as bad as it seems. Cases (i) and (ii) are the same under interchange of the primed and unprimed quantities; they describe separate states of the two systems, like the state in Figure 10; indeed, the state of Figure 10 can be accommodated within case (ii) as $d_1 = b_1$, $d_2 = \mathbf{0}$, $d_1' = \mathbf{0}$, and $d_2' = b_2$. Cases (iii)–(vi) are nothing new; they describe the same separate states as cases (i)–(ii), but in a different way. For example, in case (iii), we could replace $d_1' + d_2'$ by $d_1 + d_2 + d_1' + d_2'$, thus reducing this situation to case (i). As a specific example, the state of Figure 10 could equally well be specified by $d_1 = b_1$, $d_2 = b_2$, $d_1' = \mathbf{0}$, and $d_2' = b_2$, which falls under case (iv). Case (vii) includes all the correlated epistemic states of the two systems. For example, the correlated state of Figure 9 is specified by $d_1 = a_1$, $d_2 = a_2$, $d_1' = b_1$, $d_2' = b_2$.

Our goal now is to translate the allowed possibilities for epistemic states of two elementary systems into an efficient mathematical condition involving the toybits. To develop the right mathematical language, we define two column vectors that contain the X and Z toybits for two elementary systems:

$$\vec{a} = \begin{pmatrix} a_1 \\ a_2 \end{pmatrix}\ , \quad \vec{b} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}\ . \tag{5}$$

We then stack $\vec{a}$ on top of $\vec{b}$ to create a column vector $\vec{A}$ with four elements:

$$\vec{A} = \begin{pmatrix} \vec{a} \\ \vec{b} \end{pmatrix} \tag{6}$$

For the case of two elementary systems, we can express the two toybit constraints in terms of a 2×4 matrix, G. The toybit constraints are obtained by multiplying $\vec{A}$ by G and

equating the result to a column vector,

$$\vec{v} = \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} , \tag{7}$$

whose two elements, v_1 and v_2, give the constrained values of the appropriate linear combinations of toybits. The toybit constraints are thus expressed as

$$G\vec{A} = \vec{v} \, . \tag{8}$$

For example, the epistemic state $(1 \vee 2) \cdot (1 \vee 2)$ of Figure 10 would have the the following matrix and value vector:

$$G = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} , \quad \vec{v} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} . \tag{9}$$

The corresponding quantities for the correlated state $(1 \cdot 1) \vee (2 \cdot 2) \vee (3 \cdot 3) \vee (4 \cdot 4)$ of Figure 9 would be the following:

$$G = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix} , \quad \vec{v} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} . \tag{10}$$

In contrast, the four-box state of Figure 11, $(1 \cdot 1) \vee (2 \cdot 1) \vee (3 \cdot 2) \vee (4 \cdot 2)$, which is not an allowed epistemic state, has the following matrix and value vector:

$$G = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} , \quad \vec{v} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} . \tag{11}$$

We often find it convenient to divide our matrix G into the two parts that multiply the toybits in $\vec{a}$ and the toybits in $\vec{b}$. Both of these parts will be square matrices, and in the case of two elementary systems, they will be 2×2 matrices. We will label the left-hand side by G_X and the right-hand side by G_Z:

$$G = \left(G_X \;\middle|\; G_Z \right) . \tag{12}$$

The astute reader will recognize that the rows of G are a binary-vector representation of the quantities $d_1 + d_2$ and $d_1' + d_2'$ introduced above. Thus one can expect there to be a linear-algebraic condition for the allowed epistemic states. Indeed, by directly checking the conditions for an epistemic state listed above, we can show that a matrix G defines an allowed epistemic state if and only if its rows are linearly independent and it satisfies the *symplectic condition*

$$0 = G\Lambda G^T = G_Z G_X^T + G_X G_Z^T \, , \tag{13}$$

where

$$\Lambda = \begin{pmatrix} 0 & I \\ I & 0 \end{pmatrix} \tag{14}$$

is the fundamental (binary) symplectic matrix, which satisfies $\Lambda^2 = I$. The content of the condition $G\Lambda G^T = 0$ is that the *symplectic product* of the two rows of G must vanish, i.e.,

$$g_{1,1}g_{2,3} + g_{1,2}g_{2,4} + g_{1,3}g_{2,1} + g_{1,4}g_{2,3} = 0 \ . \tag{15}$$

Here and below we label the matrix elements of G in the standard way. The symplectic product assumes a fundamental position in the toy theory as an expression of the knowledge balance principle.

We now summarize what we have found about the epistemic states of two elementary systems.

> *Any matrix G with linearly independent rows that satisfy the symplectic condition defines an allowed epistemic state. The actual state depends on the values that are assigned to the constrained toybits; these values are contained in the vector $\vec{v}$. If we wish to talk about the Y observables, we must also specify values for the c's.*

It turns out below that nothing in this statement has to be changed for an arbitrary number of elementary systems; only the dimensions of G, $\vec{A}$, and $\vec{v}$ change.

We now want to turn to consideration of the allowed state transformations for two elementary systems. To do that, however, it is a good idea to extend our description thus far just a bit further. Given a matrix G with linearly independent rows satisfying the symplectic condition, it is always possible to find two more linearly independent rows, which we place into a 2×4 matrix

$$H = \left(H_X \ \middle| \ H_Z \right) \ ; \tag{16}$$

moreover, it is possible to choose these two rows so that H satisfies the symplectic condition, i.e., $H\Lambda H^T = 0$, and so that $G_Z H_X^T + G_X H_Z^T = I$. We can then define a 4×4 matrix $\mathcal{G}$ by stacking G on top of H:

$$\mathcal{G} = \begin{pmatrix} G \\ H \end{pmatrix} = \begin{pmatrix} G_X & G_Z \\ H_X & H_Z \end{pmatrix} \ . \tag{17}$$

This new matrix satisfies the *extended symplectic condition*:

$$\mathcal{G}\Lambda\mathcal{G}^T = \begin{pmatrix} G_Z G_X^T + G_X G_Z^T & G_Z H_X^T + G_X H_Z^T \\ H_Z G_X^T + H_X G_Z^T & H_Z H_X^T + H_X H_Z^T \end{pmatrix} = \Lambda \, . \tag{18}$$

A matrix that satisfies the extended symplectic condition we call a *symplectic matrix*. The extended symplectic condition implies that $\mathcal{G}$ is invertible (even had we not constructed it to be invertible), and its inverse is given by $\mathcal{G}^{-1} = \Lambda\mathcal{G}^T\Lambda$. That our matrix is now invertible is the reason for introducing this extension.

We now introduce a vector

$$\vec{q} = \begin{pmatrix} q_1 \\ q_2 \end{pmatrix} \, , \tag{19}$$

which contains two random bits, q_1 and q_2, by which we mean that these bits are equally likely to be 0 or 1. To put it differently, if a combination of toybits equals q_1 or q_2, this imposes no constraint on the value of that combination of toybits. Next we stack $\vec{v}$ on top of $\vec{q}$ to give a column vector

$$\vec{V} = \begin{pmatrix} \vec{v} \\ \vec{q} \end{pmatrix} \, . \tag{20}$$

With all this under our belt, we can now write the equation that defines an epistemic state as

$$\begin{pmatrix} G_X & G_Z \\ H_X & H_Z \end{pmatrix} \begin{pmatrix} \vec{a} \\ \vec{b} \end{pmatrix} = \mathcal{G}\vec{A} = \vec{V} = \begin{pmatrix} \vec{v} \\ \vec{q} \end{pmatrix} \, . \tag{21}$$

The two rows of G impose the toybit constraints that define the state. The two rows of H relate combinations of toybits to the random values q_1 and q_2 and thus impose no constraint.

This extended formulation comes up again and again throughout the following, particularly when we consider local-hidden-variable tables for epistemic states. For the present, we apply it to the allowed transformations for two elementary systems.

2. Transformations for two elementary systems

Two different epistemic states are specified by the equations $\mathcal{G}\vec{A} = \vec{V}$ and $\mathcal{G}'\vec{A} = \vec{V}'$. The vectors $\vec{V}'$ and $\vec{V}$ differ by constants in the top two entries. The matrix $\mathcal{N} = \mathcal{G}^{-1}\mathcal{G}' =$ transforms $\mathcal{G}$ into $\mathcal{G}'$ by right multiplication. It is clear that $\mathcal{N}$ is a symplectic matrix. Furthermore, if we regard transformations as being given by right matrix multiplication,

i.e., $\mathcal{G}' = \mathcal{G}\mathcal{N}$, it is easy to see that $\mathcal{N}$ must be symplectic to preserve the symplectic character of the state matrices. Only the transformation of G, i.e., the top rows of $\mathcal{G}$ affects the final state. We can conclude that an allowed state transformation is described by a symplectic matrix $\mathcal{N}$ and by two value bits that are added to $\vec{v}$.

The above is a bit of very simple linear algebra, but it enforces Spekkens's constraint that every transformation must take us to another allowed epistemic state. The right matrix multiplication by $\mathcal{N}$ performs column operations on G [7]. It turns out that all the symplectic matrices can be constructed from a small set of elementary transformations, just five in the case of 4×4 symplectic matrices:

$$\mathcal{N}_1 = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \quad \mathcal{N}_2 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}, \quad \mathcal{N}_3 = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

$$\mathcal{N}_4 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \quad \mathcal{N}_5 = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix}. \tag{22}$$

It will be easier to see how these matrices generate all of the allowed transformations on G later, after we have introduced the stabilizer formalism, as these matrices were modeled after generators for a complete set of transformations in that formalism.

Without doing the matrix multiplications, it is hard easily to see how these matrices transform the G matrix, so in the following we perform these matrix transformations on arbitrary matrices in order to illustrate how the column operations are performed. Applying $\mathcal{N}_1$ swaps the first and third columns, shown in Equation 23

$$\mathcal{N}_1 : \begin{pmatrix} g_{1,1} & g_{1,2} & g_{1,3} & g_{1,4} \\ g_{2,1} & g_{2,2} & g_{2,3} & g_{2,4} \end{pmatrix} \rightarrow \begin{pmatrix} g_{1,3} & g_{1,2} & g_{1,1} & g_{1,4} \\ g_{2,3} & g_{2,2} & g_{2,1} & g_{2,4} \end{pmatrix} \tag{23}$$

The matrix $\mathcal{N}_2$ swaps the second and fourth columns, shown in Equation 24

$$\mathcal{N}_2 : \begin{pmatrix} g_{1,1} & g_{1,2} & g_{1,3} & g_{1,4} \\ g_{2,1} & g_{2,2} & g_{2,3} & g_{2,4} \end{pmatrix} \rightarrow \begin{pmatrix} g_{1,1} & g_{1,4} & g_{1,3} & g_{1,2} \\ g_{2,1} & g_{2,4} & g_{2,3} & g_{2,2} \end{pmatrix} \tag{24}$$

The transformation $\mathcal{N}_3$ sums the first and third columns and places the result in the third column, shown in Equation 25

$$\mathcal{N}_3 : \begin{pmatrix} g_{1,1} & g_{1,2} & g_{1,3} & g_{1,4} \\ g_{2,1} & g_{2,2} & g_{2,3} & g_{2,4} \end{pmatrix} \rightarrow \begin{pmatrix} g_{1,1} & g_{1,2} & g_{1,1} + g_{1,3} & g_{1,4} \\ g_{2,1} & g_{2,2} & g_{2,1} + g_{2,3} & g_{2,4} \end{pmatrix} \tag{25}$$

The matrix $\mathcal{N}_4$ sums the second and fourth columns and places the result in the fourth column, shown in Equation 26

$$\mathcal{N}_4 : \begin{pmatrix} g_{1,1} & g_{1,2} & g_{1,3} & g_{1,4} \\ g_{2,1} & g_{2,2} & g_{2,3} & g_{2,4} \end{pmatrix} \rightarrow \begin{pmatrix} g_{1,1} & g_{1,2} & g_{1,3} & g_{1,2} + g_{1,4} \\ g_{2,1} & g_{2,2} & g_{2,3} & g_{2,2} + g_{2,4} \end{pmatrix} \tag{26}$$

The transformation $\mathcal{N}_5$ sums the second column with the first column and places the result in the second column and also replaces the third column with the sum of the third and fourth columns, shown in Equation 27

$$\mathcal{N}_5 : \begin{pmatrix} g_{1,1} & g_{1,2} & g_{1,3} & g_{1,4} \\ g_{2,1} & g_{2,2} & g_{2,3} & g_{2,4} \end{pmatrix} \rightarrow \begin{pmatrix} g_{1,1} & g_{1,2} + g_{1,1} & g_{1,4} + g_{1,3} & g_{1,4} \\ g_{2,1} & g_{2,2} + g_{2,1} & g_{2,4} + g_{2,3} & g_{2,4} \end{pmatrix} . \tag{27}$$

This sort of transformation is associated with a controlled-NOT gate in the stabilizer formalism, so we often refer to the first system as the control and the second as the target. As noted above, these five matrices can be used to general all possible symplectic transformations.

We have to remember that to specify a state transformation completely, we have to say how the value vector $\vec{V}$ changes; moreover, to interpret the Y values after the transformation, we have to know how the c's change. We want the transformations that we preform to be connected to the permutations of boxes derived by Spekkens. When we change the value vector V we must ensure that the new states we arrive at can be generally arrived at by similar permutations on the toy states. To do this transformation in such a way to generate all of the allowed permutations on toy states we must include the G matrix when we add binary vectors to the value vector. Equation 28 shows how we will preform the value vector changes.

$$G\mathcal{N} = \vec{v} + G\vec{P} \tag{28}$$

$\vec{P}$ is a binary vector with no restrictions, we multiply $\vec{P}$ with our G matrix, ot only to keep connections between states but also to ensure that we conduct allowed permutations, that do not change $\vec{P}$ for different systems. The G matrix will continue to dictate how the system evolves under transformations. We can change the c values in any way we please, but we

return to how the question of how these quantities change when we get to the stabilizer formalism below.

For the present, we consider how the formalism of a G matrix is expanded to three systems within the toy model.

C. Three elementary systems

In presenting his toy theory, Spekkens limited his discussion to one, two, and three elementary systems. It is at the three-system level that we begin to see that the knowledge balance principle yields an incomplete picture of quantum mechanics, and we must contend with situations where no matter how answers are assigned to canonical questions, we cannot mimic the predictions of quantum mechanics. Since Spekkens only considers three different three-system states, he provides very little guidance in how to use his approach for three systems and for how to associate the toy model with quantum-mechanical predictions. We are thus left pretty much to our own resources in generalizing the toy model to three systems and beyond and in developing the connections to quantum states.

The first epistemic state Spekkens considers is $(1 \vee 2) \cdot (1 \vee 2) \cdot (1 \vee 2)$. Figure 12 shows the box representation of this state. This state is one for which each system has its own, separate epistemic state, $1 \vee 2$, which satisfies the knowledge balance principle. This state is described by the toybit linear equations

$$b_1 = 0 \ , \quad b_2 = 0 \ , \quad b_3 = 0 \ . \tag{29}$$

The second three-system epistemic state that Spekkens shows us is given by $[(1 \cdot 1) \vee (2 \cdot 2) \vee (3 \cdot 3) \vee (4 \cdot 4)] \cdot (1 \vee 2)$. This is clearly the state for which the first two systems are correlated as in Figure 9, and the third system has its own separate state, $1 \vee 2$. This state is specified by the toybit constraints

$$a_1 + a_2 = 0 \ (a_1 = a_2) \ , \quad b_1 + b_2 = 0 \ (b_1 = b_2) \ , \quad b_3 = 0 \ . \tag{30}$$

Figure 13 gives us the box representation of this system.

The last state that Spekkens discusses, $(1 \cdot 1 \cdot 1) \vee (1 \cdot 2 \cdot 2) \vee (2 \cdot 1 \cdot 2) \vee (2 \cdot 2 \cdot 1) \vee (3 \cdot 3 \cdot 3) \vee (3 \cdot 4 \cdot 4) \vee (4 \cdot 3 \cdot 4) \vee (4 \cdot 4 \cdot 3)$, is one where none of the three systems has its own epistemic state. In this state, all three elementary systems are correlated. A set of toybit

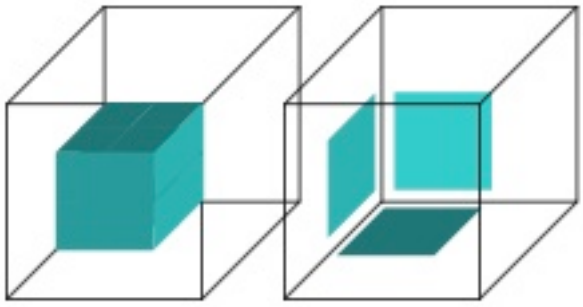

FIG. 12: The above state, $(1 \vee 2) \cdot (1 \vee 2) \cdot (1 \vee 2)$, is an allowed epistemic state for three elementary systems. Each system has its own epistemic state, in this case $1 \vee 2$, and the overall state is specified by the toybit constraints $b_1 = 0$, $b_2 = 0$, and $b_3 = 0$ Figure from [10].

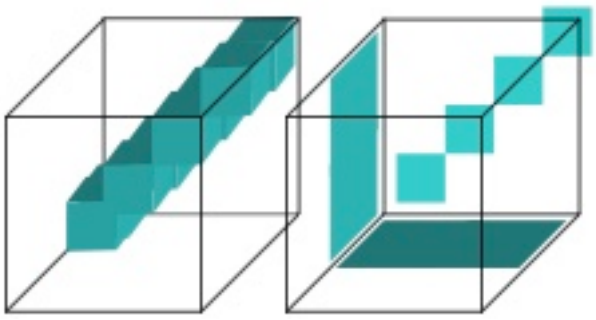

FIG. 13: Box representation of the allowed three-system epistemic state $[(1 \cdot 1) \vee (2 \cdot 2) \vee (3 \cdot 3) \vee (4 \cdot 4)] \cdot (1 \vee 2)$. The toybit constraints for this state are $a_1 + a_2 = 0$ $(a_1 = a_2)$, $b_1 + b_2 = 0$ $(b_1 = b_2)$, and $b_3 = 0$ Figure from [10].

constraints that define this state are

$$a_1 + a_2 + a_3 = 0 \ , \quad b_1 + b_2 = 0 \ , \quad b_2 + b_3 = 0 \ . \tag{31}$$

Notice that the latter two constraints require that $b_1 = b_2 = b_3$. Figure 14 gives the box representation of this state.

If we make the obvious extension of dimensions, the G matrix and value vector that describe the correlated state of Figure 14 state are

$$G = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix} \ , \quad \vec{v} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \ . \tag{32}$$

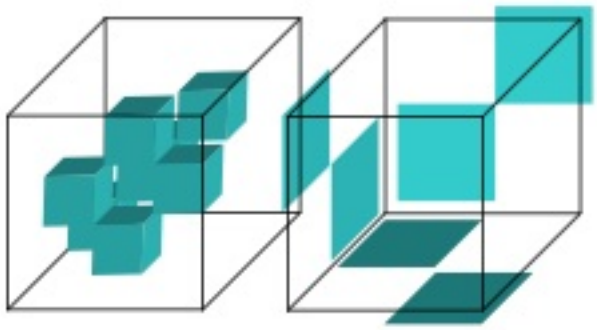

FIG. 14: Box representation of the allowed three-system epistemic state $(1 \cdot 1 \cdot 1) \vee (1 \cdot 2 \cdot 2) \vee (2 \cdot 1 \cdot 2) \vee (2 \cdot 2 \cdot 1) \vee (3 \cdot 3 \cdot 3) \vee (3 \cdot 4 \cdot 4) \vee (4 \cdot 3 \cdot 4) \vee (4 \cdot 4 \cdot 3)$. This state is specified by the toybit linear equations $a_1 + a_2 + a_3 = 0$, $b_1 + b_2 = 0$, and $b_2 + b_3 = 0$. The latter two constraints are equivalent to $b_1 = b_2 = b_3$ Figure from [10]

This matrix G has linearly independent rows and satisfies the symplectic condition, $G\Lambda G^T = 0$. We are motivated to ask if these are the general conditions for epistemic states of three systems. We can get at this result by considering the epistemic states of the first two systems as known from our previous analysis and then adding a third system. An argument just like that given above for two elementary systems establishes the conditions for a three-system epistemic state; when translated to G matrices, the conditions turn out to be that the rows of G are linearly independent and that the matrix satisfies the symplectic condition. Indeed, the italicized state statement in Section I.B.1 applies without change to three elementary systems.

In the same way, we can also generalize our discussion of allowed transformations to three elementary systems: the allowed transformations are described by symplectic matrices. This suggests that we now have the resources to generalize the toy model to an arbitrary number of systems, and this we now show.

D. N elementary systems

1. *Epistemic states for N elementary systems*

A combination of N elementary systems requires the knowledge of $2N$ bits of information to fully identify a particular ontic state. Knowledge of N bits is necessary to specify

an epistemic state of maximal knowledge, but as we have already seen for two and three elementary systems, the N bits must be chosen so as not to violate the knowledge balance principle for any subset of the N systems.

To depict the epistemic states of N systems in terms of Spekkens's boxes would require an N-dimensional picture, which means that pictures run out of gas after three systems. Instead, we will specify N-system states in terms of their G matrices and the value vector $\vec{v}$. To consider the values of the Y observables of the N systems, we must also specify a value of c for each system. At this point, it should not be surprising that the requirement on the G matrices is that they satisfy the symplectic condition. Indeed, the italicized statement in Section I.B.1 remains valid for N systems. To show this, we can consider an inductive proof that assumes the statement is valid for $N-1$ systems and then uses an argument identical to that in Section I.B.1 to show the statement is valid for N systems. In just the same way as we did for two elementary systems, the G matrix can be extended to an invertible $\mathcal{G}$ matrix by stacking the G matrix on top of an associated matrix H.

2. Transformations for N elementary systems

The allowed transformations for N elementary systems are described by multiplying the state matrix G on the right by matrices $\mathcal{N}$ of size $2N \times 2N$. Using the same argument we used for two and three elementary systems, we find that the allowed transformation matrices are symplectic transformations. To describe a state transformation fully, of course, we must also specify what happens to the value vector $\vec{v}$.

The set of $2N \times 2N$ symplectic matrices might seem to be quite a complicated set to describe, but in fact, all such matrices are generated by matrices of the form discussed in Section I.B.2, but applied to any pair of systems among the N available. To see how this works out, we introduce a new notation that refers to the X sector of the G matrix by a set of column vectors, $r_a(n)$, where the index n refers to the n^{th} column of the X sector. Similarly, the Z sector will be referred to by $r_b(n)$. We consider now three classes of transformations. The first class of transformations contains the ones much like the $\mathcal{N}_1$ and $\mathcal{N}_2$ transformations at the two-system level. The transformations in this first class can be defined by explaining

how the final G matrix will result from the initial matrix:

$$\begin{aligned} r_a(n) &\rightarrow r_b(n) \ , \\ r_b(n) &\rightarrow r_a(n) \ . \end{aligned} \tag{33}$$

Transformations of this first class affect only one elementary system, swapping the n^{th} columns of the X and Z sectors. Figure 15 shows this column swap in detail.

$$\left(\begin{array}{ccccc|ccccc} g_{1,1} & \cdots & g_{1,n} & \cdots & & \cdots & g_{1,N+n} & \cdots & g_{1,2N} \\ g_{2,1} & \cdots & g_{2,n} & \cdots & & \cdots & g_{2,N+n} & \cdots & g_{2,2N} \\ \vdots & \ddots & \vdots & \ddots & & \ddots & \vdots & \ddots & \vdots \\ g_{N,1} & \cdots & g_{N,n} & \cdots & & \cdots & g_{N,N+n} & \cdots & g_{N,2N} \end{array}\right) \rightarrow \left(\begin{array}{ccccc|ccccc} g_{1,1} & \cdots & g_{1,N+n} & \cdots & & \cdots & g_{1,n} & \cdots & g_{1,2N} \\ g_{2,1} & \cdots & g_{2,N+n} & \cdots & & \cdots & g_{2,n} & \cdots & g_{2,2N} \\ \vdots & \ddots & \vdots & \ddots & & \ddots & \vdots & \ddots & \vdots \\ g_{N,1} & \cdots & g_{N,N+n} & \cdots & & \cdots & g_{N,n} & \cdots & g_{N,2N} \end{array}\right)$$

FIG. 15: Effect on G of a matrix in the first class of transformations. On the left-hand side of the arrow we have the original matrix with the n and $n+N$ columns highlighted. This transformation interchanges these two columns. The matrix on the right-hand side shows this interchange explicitly by swapping the colors of the highlighted columns.

The second class of transformations are the ones much like the $\mathcal{N}_3$ and $\mathcal{N}_4$ transformations at the two-system level:

$$r_b(n) \rightarrow r_b(n) + r_a(n) \ . \tag{34}$$

This second class of transformations is portrayed in Figure 16, where the two columns in question are highlighted in different colors. As for the first class, these transformations only affect a single elementary system. The column addition is shown as the combination of the two highlighted colors.

The last class of transformations are the ones much like the $\mathcal{N}_5$ transformations at the two-system level. These transformations affect two elementary systems, a control, n, and a target, m:

$$\begin{aligned} r_a(m) &\rightarrow r_a(m) + r_a(n) \\ r_b(n) &\rightarrow r_b(m) + r_b(n) \end{aligned} \tag{35}$$

Figure 17 shows this last class of transformations, again with the highlighted columns in question and the addition of these columns regarded as the combinations of the highlight colors. With these three classes of transformations, we can build any allowed transformation for an arbitrary N dimensional system [10].

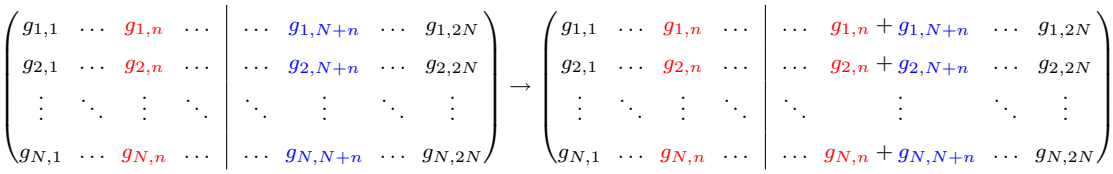

FIG. 16: Effect on G of a matrix in the second class of transformations. On the left-hand side of the arrow we have the original matrix with the n and $n+N$ columns highlighted. During this transformation the n column is added to the $n+N$ column. The matrix on the right hand highlights that these two columns have been summed.

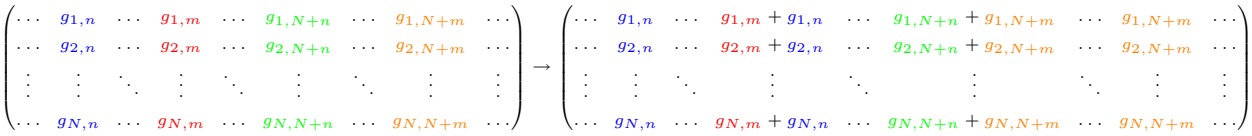

FIG. 17: Effect on G of a matrix in the second class of transformations. On the left-hand side of the arrow we have the original matrix with the four columns highlighted: n (blue), m (red), $N+n$ (green) and $N+m$ (orange). During this transformation the n column is added to the m column and the $m+N$ column is added to the $n+N$ column. The matrix on the righthand side shows the two combined columns in both the m and $N+n$ columns with the colors of the $m+N$ and n columns remaining unchanged.

II. STABILIZER FORMALISM

The *stabilizer formalism* is a well known theoretical apparatus used to describe the states of a two-level system, or *qubit*, that lie on the axes of the Bloch sphere. The Bloch sphere, shown in Figure 18, is much like the Poincaré sphere, but for electron spin rather then for polarization of light. We must begin our discussion of the Stabilizer formalism by introducing the *Pauli group* under matrix multiplication. The Pauli matrices (and the identity matrix) are normally denoted by σ_α where the subscript $\alpha = 0, 1, 2, 3$ or rather $\alpha = 0, x, y, z$. From this point on, we will denote the Pauli matrices by the capital letters I, X, Y, Z, where $I = \sigma_0$ is the identity matrix, and we will only revert to the σ notation when referring to arbitrary Pauli matrices:

$$X = \sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \quad , \quad Y = \sigma_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \quad , \quad Z = \sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \quad . \tag{36}$$

The Pauli group P for a single qubit contains the Pauli matrices along with the freedom to multiply them by ± 1 and $\pm i$,

$$P = \{\pm I \ \pm iI, \pm X, \pm iX, \pm Y, \pm iY, \pm Z, \pm iZ\} \ . \tag{37}$$

We will often be considering a system of N qubits, and then the Pauli groups consists of all N-fold tensor products of Pauli matrices along with the freedom to multiply them by ± 1 and $\pm i$.

A. Stabilizer states

We say a state is stabilized by a (Hermitian) operator when that state is a $+1$ eigenstate of that operator. For example, the state $|0\rangle$ is stabilized by the Pauli matrix Z, i.e., $Z|0\rangle = |0\rangle$, and the state $|1\rangle$ is stabilized by $-Z$, i.e., $-Z|1\rangle = -|1\rangle$. In this way we can deal with measurements that will yield a $+1$ value, instead of with the actual state. For a system of N qubits, the stabilized states are $+1$ eigenstates of tensor products of N Pauli operators, e.g.,

$$Z_1 \otimes Z_2 \otimes \ldots \otimes Z_N |0\rangle_1 \otimes |0\rangle_2 \otimes \ldots \otimes |0\rangle_N = |0\rangle_1 \otimes |0\rangle_2 \otimes \ldots \otimes |0\rangle_N \ . \tag{38}$$

We will generally not use the tensor-product symbol between the kets or the operators, using the subscripts on the operators to identify which system they operate on, and ordering the kets correctly. Using this simplified notation, Eq. (38) would reduce to

$$Z_1 Z_2 \ldots Z_N |00\ldots0\rangle = |00\ldots0\rangle \ . \tag{39}$$

Often it is convenient to streamline the notation even further by omitting the subscripts on the Pauli operators in a tensor-product string, since the ordering of the operators tells which system an operator acts on. Thus, for example, we could write for three qubits, $Z_1 Z_2 Z_3 = ZZZ$, or for five qubits, $X_2 Y_4 = IXIYI$.

A stabilizer state of N qubits will have 2^N (commuting, Hermitian) stabilizers; these are said to make up the (abelian) *stabilizer group*. We only need to specify a set of N independent stabilizer generators from the group in order to generate the complete group of 2^N stabilizers to fully identify a particular state. We are interested in the particular case of N (independent and commuting) generators that have a specific simultaneous $+1$ eigenstate.

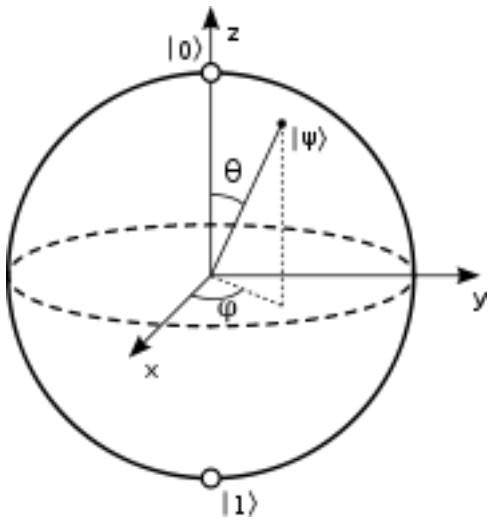

FIG. 18: Bloch sphere. The spin up and down states, which we call $|0\rangle$ and $|1\rangle$, lie on the $\pm z$-axis. Equal superpositions of spin up and down lie in the equatorial plan. The states on the $\pm x$-axis are $|\pm\rangle = (|0\rangle \pm |1\rangle)/\sqrt{2}$, and the states on the $\pm y$-axis are $|\pm i\rangle = (|0\rangle \pm i|1\rangle)/\sqrt{2}$.

We can make a connection with the toy theory by defining the same sort of matrix G as we did for toy theory. First we need to represent Pauli matrices with binary numbers, so we define two bits $r_a(\alpha)$ and $r_b(\alpha)$, where $\alpha = 0, 1, 2, 3$ is a label on a Pauli operator:

$$r_a(\alpha) = \delta_{x,\alpha} + \delta_{y,\alpha} = \begin{cases} 0\ , & \alpha = I \\ 1\ , & \alpha = X \\ 0\ , & \alpha = Z \\ 1\ , & \alpha = Y \end{cases} ,$$

$$r_b(\alpha) = \delta_{z,\alpha} + \delta_{y,\alpha} = \begin{cases} 0\ , & \alpha = I \\ 0\ , & \alpha = X \\ 1\ , & \alpha = Z \\ 1\ , & \alpha = Y \end{cases} .$$

Turning this labeling around, we can now designate a Pauli operator by the corresponding binary number $r_a r_b$, i.e., $\sigma_0 = I = \sigma_{00}$, $\sigma_3 = Z = \sigma_{01}$, $\sigma_1 = X = \sigma_{10}$, and $\sigma_2 = Y = \sigma_{11}$.

Next we use the fact that $Y = iXZ$ and our definition of the binary specifier to represent an arbitrary Pauli operator as

$$\sigma_{r_a r_b} = i^{r_a r_b} X^{r_a} Z^{r_b} \ . \tag{40}$$

We would now like to write an arbitrary generator for an N qubit system in this way. We can do this by taking an N-fold tensor product of our arbitrary Pauli operator:

$$\sigma_{r_a(1)r_b(1)} \otimes \cdots \otimes \sigma_{r_a(N)r_b(N)} = \bigotimes_{k=1}^{N} i^{r_a(k)r_b(k)} X^{r_a(k)} Z^{r_b(k)} \ . \tag{41}$$

We can now pull out the i's and get that any N-fold tensor product—and, hence, any generator, up to a preceding sign of ± 1—is given by

$$\sigma_{r_a(1)r_b(1)} \otimes \cdots \otimes \sigma_{r_a(N)r_b(N)} = i^{\sum_{k=1}^{N} r_a(k)r_b(k)} \bigotimes_{k=1}^{N} X^{r_a(k)} Z^{r_b(k)} \ . \tag{42}$$

We can now fully specify a generator, up to a sign, by a $2N$-dimensional row vector, called the *check vector*, which contains all the binary specifiers:

$$\vec{r} = \left(\begin{array}{c|c} \vec{r}_a & \vec{r}_b \end{array} \right) = \left(\begin{array}{c|c} r_a(1) \ \cdots \ r_a(N) & r_b(1) \ \cdots \ r_b(N) \end{array} \right) \ . \tag{43}$$

Key facts are, first, that two Pauli products commute if and only if the symplectic product of their check vectors, $\vec{r}_1$ and $\vec{r}_2$, is zero, i.e.,

$$0 = \sum_{k=1}^{N} r_{1a}(k) r_{2b}(k) + r_{1b}(k) r_{2a}(k) \ , \tag{44}$$

and, second, that two generators are independent if and only their check vectors are linearly independent. [3]

If we have N (independent and commuting) generators, we can fully specify them by their check vectors $\vec{r}_1, \ldots, \vec{r}_N$, which are linearly independent and have vanishing pairwise symplectic products. Thus, if we stack the check vectors on top of one another as the rows of a $N \times 2N$ matrix

$$G = \begin{pmatrix} \vec{r}_1 \\ \vec{r}_2 \\ \vdots \\ \vec{r}_N \end{pmatrix} \ , \tag{45}$$

this matrix G has linearly independent rows and satisfies the symplectic condition. We call such a matrix a *generator matrix*;we realize now that the G matrices of the toy model are such generator matrices.

Since the generator matrix only represents the stabilizer generators up to a sign, we should regard the generator matrix as specifying not a particular state, but rather as specifying an

orthonormal basis of states, with the signs of the generators giving the ± 1 eigenvalues of the Pauli products. The 2^N possible strings of eigenvalues correspond precisely to the 2^N vectors in an orthonormal basis; they can alternatively be regarded as the values contained in the vector $\vec{v}$ of the toy model. Having established this correspondence, we now refer to G in both the toy model and the stabilizer formalism as the generator matrix, and we refer to $\vec{v}$ in both situations as the value vector.

We will now have to consider matrices that transform between stabilizer states [1, 4, 5, 9].

B. The normalizer group and stabilizer transformations

Elements of the normalizer group are transformations that unitarily conjugate Pauli-group elements, i.e., tensor products of Pauli operators, to other Pauli-group elements. For instance, for a single qubit, the Hadamard gate, H, which is defined as,

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} = H^\dagger \ . \tag{46}$$

alters the Pauli matrices in the following way:

$$\begin{aligned} H^\dagger X H &= Z \ , \\ H^\dagger Z H &= X \ , \\ H^\dagger Y H &= -Y \ . \end{aligned} \tag{47}$$

[9] [4] [5]

Since a normalizer element will conjugate a generator set to another generator set, it can be said to transform the corresponding generator matrix. The transformation of the generator matrix will be described by multiplying the generator matrix on the right by a symplectic matrix, so there is a one-to-one correspondence between normalizer elements and symplectic transformations of the toy theory. To describe a state transformation in the toy model completely, however, we need also to say how the value vector transforms. The toy model, as a hidden-variable model, is not equivalent to quantum mechanics. Thus there is generally no value vector that duplicates the predictions of quantum mechanics. We return to this disconnect between the toy model and quantum mechanics below.

We can generate the entire group of normalizer elements for N qubits using just two single-qubit operations and an entangling operation. [1] We use the Hadamard and the S

gate for our single-qubit operations, and the controlled-NOT for our entangling operation. The S gate, sometimes called the phase gate, is defined by

$$S = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix} \ ; \tag{48}$$

it conjugates the Pauli matrices in the following way:

$$\begin{aligned} S^\dagger X S &= Y \ , \\ S^\dagger Y S &= -X \ , \\ S^\dagger Z S &= Z \ . \end{aligned} \tag{49}$$

Because the controlled-NOT (CN) is a two-qubit operation, we invent a notation to specify which qubit we are controlling and which qubit is our target. We will use the notation $n \to m$ to designate that qubit n is our control and qubit m is our target. The matrix representation of the CN can be written as

$$\mathrm{CN}_{1\to 2} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \ . \tag{50}$$

I find this matrix representation to be less enlightening than displaying how the basis states are transformed using the controlled-NOT operation:

$$\begin{aligned} \mathrm{CN}_{1\to 2}|00\rangle &\to |00\rangle \ , \\ \mathrm{CN}_{1\to 2}|01\rangle &\to |01\rangle \ , \\ \mathrm{CN}_{1\to 2}|10\rangle &\to |11\rangle \ , \\ \mathrm{CN}_{1\to 2}|11\rangle &\to |10\rangle \ . \end{aligned} \tag{51}$$

We can quickly determine how the controlled-NOT conjugates Pauli products:

$$\begin{aligned} \mathrm{CN}^\dagger_{n\to m} I_n X_m \mathrm{CN}_{n\to m} &= I_n X_m \ , \\ \mathrm{CN}^\dagger_{n\to m} X_n I_m \mathrm{CN}_{n\to m} &= X_n X_m \ , \\ \mathrm{CN}^\dagger_{n\to m} I_n Z_m \mathrm{CN}_{n\to m} &= Z_n Z_m \ , \\ \mathrm{CN}^\dagger_{n\to m} Z_n I_m \mathrm{CN}_{n\to m} &= Z_n I_m \ . \end{aligned} \tag{52}$$

Using these conjugations and that $Y = iXZ$, we can derive how all other Pauli products are conjugated. [9] [4] [5]

These three operations, H and S on all qubits and controlled-NOT on all pairs of qubits, generate the normalizer group and thus can be used to transform from any stabilizer state to any other stabilizer state. These transformations correspond to the classes of symplectic transformations within the toy model that were discussed in Sections I.B.2 and I.D.2. That these transformations generate the normalizer group shows that the corresponding symplectic transformations generate all symplectic transformations within the toy model. Although the normalizer elements are in correspondence with symplectic transformations of the generator matrix, to describe a transformation in the toy model completely also requires knowing how the value vector changes and how the c values change. We can get an idea of what this means for quantum mechanics by examining the H, S, and controlled-NOT transformations more closely.

The Hadamard gate applied to the to the n^{th} system swaps X and Z for that system, while leaving Y the same, and thus clearly corresponds to the symplectic transformation of Figure 15, which swaps columns n and $N + n$ of the generator matrix. This swap does not, however, take into account the change in sign of Y that accompanies a Hadamard gate. To account for this, we could imagine flipping c_n, i.e., taking c_n to $c_n + 1$ as part of the transformation. With this change, we have fully accounted for all the effects of a Hadamard gate.

An S gate on the n^{th} system conjugates X to Y and Y to $-X$, while leaving Z unchanged. It thus corresponds to the symplectic transformation of Figure 16, except that we need to take into account the sign change when Y goes to $-X$. To do this, we need to correctly choose the $\vec{P}$ of Equation 28, while adjusting the c values so we get the sign change only when a state is stabilized by a Y observable.

The situation with the controlled-NOT is yet more complicated. We can account for the controlled-NOT transformations (50) by using the symplectic transformation of Figure 17. The problem arises because the Pauli operators are anti-commuting quantities, satisfying $XYZ = i$, whereas the toy observables are commuting quantities, satisfying $XYZ = W =$

$(-1)^c$. The troubling effects show up in the transformations of $X_n Z_m$ and $Y_n Y_m$:

$$\begin{aligned} \mathrm{CN}^\dagger_{n\to m} X_n Z_m \mathrm{CN}_{n\to m} &= X_n Z_n X_m Z_m = -Y_n Y_m \ , \\ \mathrm{CN}^\dagger_{n\to m} Y_n Y_m \mathrm{CN}_{n\to m} &= -X_n Z_m \ . \end{aligned} \tag{53}$$

Just as for the phase gate, we believe that we could account for the negative signs here by making flipping entries in the $\vec{P}$ vector and changing the c values in a nonlinear way.

III. LOCAL HIDDEN VARIABLES

A. Local-hidden-variable tables

The theory of local hidden variables (LHV) was proposed as a way of explaining the probabilistic predictions of quantum mechanics as coming from the effects of unknown hidden variables. It is known, however, that if the hidden variables are properties of local quantum systems, they cannot account for all the predictions of quantum mechanics when the systems are entangled. Thus entanglement is often thought of as nonlocal, or quantum, correlations.

In an LHV model, we suppose that a quantum system has a particular hidden state—an ontic state in the terminology used here—but that we have limited information about the ontic state and thus are restricted to making predictions based on that limited information. The state of limited information we are calling an epistemic state. For example, in the present context, we would say that an elementary two-level system has three spin components, x, y, and z, but we cannot have complete information about all these components. An LHV table is a chart that summarizes what we know about the three components of spin; it uses random bits to represent what we do not know. For example, the two-qubit quantum state $|00\rangle$ could be described by the LHV table shown in Table 1. This table clearly shows all of the possible results if we measured X, Y or Z on either qubit. Obviously we would get a $+1$ value if we measured either IZ, ZI or ZZ, but all other measurements, besides the identity on both qubits, would yield $+1$ or -1 with equal probability and are treated as random answers. This table also clearly identifies the state $|00\rangle$ and none other. There is no other state that could correctly have the measurement results predicted in this particular LHV table.

The LHV tables we consider here require three bits of information for each elementary system, one for each spin component. Of the $3N$ bits for N elementary systems, N bits

TABLE I: Hidden Variable table for $|00\rangle$

Qubit	X	Z	Y
1	$(-1)^{q_1}$	1	$(-1)^{q_1}$
2	$(-1)^{q_2}$	1	$(-1)^{q_2}$

specify the correlation of the three spin components for each elementary system, i.e., $XYZ = W = (-1)^c$. The other $2N$ bits would be necessary to specify all spin components, i.e., to specify an ontic state, but we only allow knowledge of N of these $2N$ bits, in accordance with Spekkens's knowledge balance principle. Thus our LHV tables are another representation of the epistemic states in the toy model, but with the advantages that the random variables and the correlations among the three spin components are explicitly displayed.

We now formulate the construction of LHV tables in detail. The measurement outcomes for the three spin components can be either ± 1, so we write the measurement outcomes of X and Z and their connections with Y as

$$X_j = (-1)^{a_j} \ , \quad Z_j = (-1)^{b_j} \ , \quad X_j Y_j Z_j = (-1)^{c_j} \ , \tag{54}$$

where a_j, b_j, and c_j are binary digits for the j elementary system, the toybits of Spekkens's model. The N toybit constraints are contained in the generator matrix G. As we discussed in Section I.B, we can expand the generator matrix to a symplectic matrix $\mathcal{G}$, by adding N linearly independent rows, and we can expand the value vector $\vec{v}$ to a vector $\vec{V}$ by adding N random bits in a vector $\vec{q}$. Thus the toybits contained in the vector $\vec{A}$ are constrained by

$$\mathcal{G}\vec{A} = \vec{V} \ . \tag{55}$$

Using the inverse of $\mathcal{G}$, we can determine $\vec{A}$:

$$\begin{pmatrix} \vec{a} \\ \vec{b} \end{pmatrix} = \vec{A} = \mathcal{G}^{-1}\vec{V} = \Lambda \mathcal{G}^T \Lambda \vec{V} = \begin{pmatrix} H_Z^T & G_Z^T \\ H_X^T & G_X^T \end{pmatrix} \begin{pmatrix} \vec{v} \\ \vec{q} \end{pmatrix} = \begin{pmatrix} H_Z^T \vec{v} + G_Z^T \vec{q} \\ H_X^T \vec{v} + G_X^T \vec{q} \end{pmatrix} \ . \tag{56}$$

This result determines the X and Z columns of the LHV table, and the Y column can be filled in by using the assumed correlations of X, Y, and Z, i.e., $Y_j = (-1)^{c_j} X_j Z_j = (-1)^{a_j + b_j + c + j}$.

We emphasize again that the LHV tables are another, perhaps more complete, way of representing the results of Spekkens's toy theory. Instead of using the N equations involving $\vec{a}$ and $\vec{b}$ to constrain the toybits and reduce the number of shaded boxes to 2^N from the total

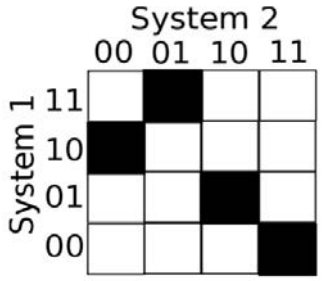

FIG. 19: The above state is associated with the linear equations, $a_1 + a_2 + b_1 = 1$, $b_1 + b_2 = 1$ and $c_1 + c_2 = 0$.

number of boxes, 4^N, we here find $\vec{a}$ and $\vec{b}$ in terms of N bits we are allowed to know and N random placeholders, the q's. Changing the signs of the values in $\vec{v}$ or changing the c values generally forces some predictions to change signs. [3] [2]

B. Examples of local-hidden-variable tables

The linear algebra and notation can get confusing, I think that it is important to work out an example and to see these connections directly. Let's examine the system $|\psi\rangle = \frac{1}{\sqrt{2}}(|01\rangle + i|10\rangle)$. The stabilizers for this system are $\{YX, -XY, -ZZ, II\}$, this can be easily verified. The generator matrix for this system is,

$$G = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix}. \tag{57}$$

We can also write down the vector $\vec{v}$ to help us derive our toy theory picture. Equation 58 can be used to derive this visual representation,

$$\begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ b_1 \\ b_2 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}. \tag{58}$$

Figure 19 shows us our toy model representation of the state.

We now have the toy model version and the stabilizer formalism model for the same state. Let's see if we can figure out the LHV table for this same state. We add on our duel

generator matrix making our complete generator matrix,

$$\mathscr{G} = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}. \tag{59}$$

After our linear algebra to derive our local hidden variable tables we end up with

$$\begin{pmatrix} a_1 & a_2 & b_1 & b_2 \end{pmatrix} = \begin{pmatrix} v_1 + c_1, & v_2, & q_1 + c_2, & q_2 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \end{pmatrix}, \tag{60}$$

therefore,

$$\begin{aligned} a_1 &= v_1 + q_1 + r_2, & a_2 &= q_2, \\ b_1 &= q_1 + c_1, & b_2 &= v_2 + q_1 + c_1. \end{aligned} \tag{61}$$

Remembering that $X_n Y_n Z_n = (-1)^{c_n}$ our local hidden variable table is shown in Table 2.

TABLE II: Hidden Variable table for $|\psi\rangle = \frac{1}{\sqrt{2}}(|01\rangle + i|10\rangle)$

Qubit	X	Z	Y
1	$(-1)^{v_1+q_1+q_2}$	$(-1)^{q_1+c_1}$	$(-1)^{v_1+q_2}$
2	$(-1)^{q_2}$	$(-1)^{v_2+q_1+c_1}$	$(-1)^{q_2+v_2+c_1+c_2+q_1}$

To conclude that we have the same state we have to correctly choose elements of $\vec{v}$ and our c values so that we get the correct measurement result for the stabilizers, i.e. in this case we need to get -1 for XY and so on. To do this we must choose $v_1 = 0$, $v_2 = 1$, and $c_1 + c_2 = 0$. Our final LHV table is shown in Table 3. How about the transformations? Let's do an example of a transformation so we can examine how each of the states change. We will try preforming the normalizer $H_1 S_2$. Performing this transformation on the stabilizers we find that our new stabilizers are $\{-YX, -ZZ, XY, II\}$. Sparing the reader the details in the linear algebra, the associated LHV table for these stabilizers is given in Table 4.

TABLE III: Hidden Variable table for $|\psi\rangle = \frac{1}{\sqrt{2}}(|01\rangle + i|10\rangle)$

Qubit	X	Z	Y
1	$(-1)^{q_1+q_2}$	$(-1)^{q_1+c_1}$	$(-1)^{q_2}$
2	$(-1)^{q_2}$	$(-1)^{1+q_1+c_1}$	$(-1)^{q_2+1+q_1}$

TABLE IV: Hidden Variable table for our new transformed state

Qubit	X	Z	Y
1	$(-1)^{1+q_1+q_2}$	$(-1)^{q_1+c_1}$	$(-1)^{1+q_2}$
2	$(-1)^{q_2}$	$(-1)^{q_1+1+c_1}$	$(-1)^{q_2+1+q_1}$

If we instead do the transformations on the generator matrices we get $\mathscr{G}$ and also the changes in the c and $\vec{v}$ values all given by,

$$\mathscr{G} = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0. \end{pmatrix}, c_1 + c_2 = 1 \text{ and } \vec{v} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}. \tag{62}$$

We can find the LHV table here to make sure that we obtain the same answer as before. For this full generator matrix we get our LHV table shown in Table 5. This table appears to be

TABLE V: Hidden Variable table for our new transformed state

Qubit	X	Z	Y
1	$(-1)^{q_1+c_1}$	$(-1)^{q_1+q_2+1}$	$(-1)^{q_2+1}$
2	$(-1)^{q_2+c_2}$	$(-1)^{q_1+q_2}$	$(-1)^{q_1}$

different from table 4 but in reality it is the same. LHV tables are the same when they have the same predictions for the measurements that they can predict and the have uncertain measures for everything else. These two tables have that quality; if we just examine the instances where we get the random variables to cancel each other, we can quickly see that we arrive at the correct measurement outcomes that we desire. Finally Figure 20 shows the transformed toy theory box.

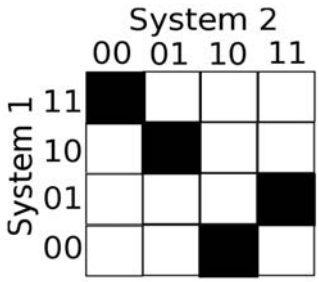

FIG. 20: The above state is associated with the linear equations, $a_1 + a_2 + b_1 = 0$, $b_1 + b_2 = 1$ and $c_1 + c_2 = 1$.

IV. CONCLUSIONS

The generator matrices are the common element in the toy model, the stabilizer formalism, and the LHV tables. All of these descriptions use the same generator matrices, and all of the transformations within the three descriptions are given in terms of symplectic matrices. In fact, the elementary transformations on the generator matrices, corresponding to the Hadamard gate, the S gate, and the controlled-NOT, were selected because it was already known that they generate the normalizer group for the stabilizer formalism.

Each of these descriptions satisfies the knowledge balance principle. The box representation of the toy theory is based on satisfying a set of N linearly independent equations. If we complete these N equations with N additional equations, which since they involve random bits, do not constrain the values of toybits, we find the LHV tables. For both the toy theory and the LHV tables, it is necessary to specify an additional bit for each elementary system, which determines how to assign the value of the Y observable for that elementary system. Since the LHV tables directly display all of this information, they are more informative than Spekkens's box representation.

Quantum mechanics is different in that no matter what value vector is chosen for a generator matrix, i.e., for a stabilizer state, and what c bits are chosen, it is generally not possible for the corresponding LHV table to mimic all of the predictions of quantum mechanics for the stabilizer state under consideration. We are left, in this context, with the familiar, but still puzzling conclusion that quantum mechanics cannot be fully described by any local-hidden-variable model.

Appendix A: All of the group theory needed for this paper

Group theory has an essential role in our work and I think that it is important to have a quick review. A group $\{S\}$ is a set of elements with an elementary product "$\cdot$", such as addition, scalar multiplication, or matrix multiplication such that for any group elements s_1 and s_2 the following holds:
(i) For all s_1, s_2 in S we have $s_1 \cdot s_2 = s_3$ such that $s_3 \in S$; this ensures that the group is closed.
(ii) There exists an element e in S such that for all s in S, $e \cdot s = s \cdot e = s$ holds; this ensures the existence of an identity.
(iii) There exists a s_1^{-1} in the group S such that $s_1^{-1} \cdot s_1 = e = s_1 \cdot s_1^{-1}$ where e is the identity and is contained in the group S; this ensures the existence of an inverse element.
(iv) For all s_1, s_2, and s_3 in S $(s_1 \cdot s_2) \cdot s_3 = s_1 \cdot (s_2 \cdot s_3)$ ensures the associativity of group elements.

We also need to know a little about group generators. Elementary products of generators will yield all other group members, we say that a group is "generated" by this set of members when this condition holds. Consider s_1 and s_2, elements of group S. We say these two elements generate the group S if for every element h in S, $h = s_1^{f_1} \cdot s_2^{p_1} \cdot \ldots s_1^{f_n} \cdot s_2^{p_n}$ where f_n and p_n are numbers.

We will be concerned with the Pauli group throughout this paper. The Pauli group is the group of Pauli matrices, along with the matrix product,

$$P = \{\pm I \ \pm iI, \pm X, \pm iX, \pm Y, \pm iY, \pm Z, \pm iZ\}. \tag{A1}$$

We use the Pauli group in this paper to create stabilizer elements for systems of N-qubits by taking a N-fold tensor product of elements of the Pauli group. The Pauli group is generated by X and Z. [9] [8] [6]

Appendix B: Proof 1

We would like to prove that by simply defining N generators for the group of 2^N stabilizers specifying a particular state demands that the generators commute with each other. It is easy to see that $-I$ is not a stabilizer, there is no state $|\psi\rangle$ such that $-I|\psi\rangle = +|\psi\rangle$.

All elements of the Pauli group either commute or anti-commute, therefore every stabilizer generator must commute or anti-commute. Let's assume that two generators for an N system anti-commute, i.e. $s_1 s_2 = -s_2 s_1$. By assumption we know that both s_1 and s_2 are stabilizers, i.e,

$$\begin{aligned} s_1|\psi\rangle &= +|\psi\rangle, \\ s_2|\psi\rangle &= +|\psi\rangle. \end{aligned} \tag{B1}$$

We also know that any matrix product of any arbitrary generators must be a stabilizer of the same state., i.e,

$$\begin{aligned} s_1 s_2|\psi\rangle &= +|\psi\rangle, \\ s_2 s_1|\psi\rangle &= +|\psi\rangle. \end{aligned} \tag{B2}$$

Our assumption requires $-s_2 s_1$ to be a stabilizer, i.e. $-s_2 s_1|\psi\rangle = +|\psi\rangle$ must be true and thus $-I|\psi\rangle = +|\psi\rangle$. We have arrived at a contradiction, since we know that $-I$ is not a stabilizer. Therefore our assumption must be false, and all generators for a stabilizer state must commute.[9] [4] [5]

Appendix C: Proof 2

We would like to show that simply using Spekkens's knowledge balance principle requires that generators of his toy bits commute with one another, i.e $[s_1, s_2] = 0$. We plan to prove this by induction, beginning the proof at the first non trivial case, as it is easy to see for an elementary system where only one toy bit of information can be known. For the following proof we will say that X and Z anti-commute in his toy theory.

For a system of two states we will examine the commutation condition in parts. Let us first assume that s_1 has toy bit corresponding to the identity I for the first state of the two. We can assume this without loss of generality because the same thing could be done for the second state and the second generator. The first generator is now $s_1 = IA$ where A can be X, Y, or Z. If we take $A = I$ we get the identity on both states, this is the trivial solution of all systems with two states and will commute with everything. We will call $s_2 = BC$ where we already know that B commutes with I in the first generator. We have two cases, case 1 $[A, C] = 0$ in this case we are done and both generators commute with each other. In case

2, $[A, C] \neq 0$ in this case $A \neq C$ and therefore we violate the knowledge balance principle because whatever the value of the toy bit A we can infer the value of the toy bit C and have total knowledge of the second system. Therefore $[s_1, s_2] = 0$.

Let's expand this to the case where there is no explicit I in either generator. In this instance we have three cases. The first case is where both of the generators commute, this is what we are trying to prove so if this is the case we are done. Case two is when we know that one of the bits for the two systems commutes. Since we have already done the identity case we will examine the only other possible case here when the toy bit for this system is the same for both generators. We have, $s_1 = AB$ and $s_2 = AC$. If B and C commute we are done; if they don't commute it means that again they are completely different toy bits and we again violate the knowledge balance principle for the second system. The last case is where the first system of both generators anti-commute. If the toy bits for the second system commute we again have the problem where we could extrapolate more information about the first system then the knowledge balance principle allows and thus is not allowed in Spekkens's model. The only other case is where the toy bits of the second system also anti-commute demanding the commutation of the generators. Therefore all generators for two states must commute to correctly satisfy the knowledge balance principle.

We assume that the same holds for the N system case and examine the $N + 1$ system. If the first N systems commute we can assume that the first N systems are an allowed state of the knowledge balance principle and the new $N + 1$ system must be some sort of product state. Therefore the last system must commute within all of the generators or we would be adding two bits of information about the last system. If the first N systems of the generators do not commute then by our induction assumption we conclude that the first N systems cannot stand alone as a valid toy system, therefore the first N systems must be entangled with the last system we added on. This can only be accomplished when the toy bit for the last system anti-commutes within these generators. Therefore, by induction, when we use the knowledge balance principle we enforce all generators for a particular system to commute with one another.

[1] Brian Eastin. Error channels and the threshold for fault-tolerant quantum computation. Ph.D. thesis, arXiv:0710.2560, 2007.

[2] Albert Einstein, Boris Podolsky, and Nathan Rosen. Can a quantum-mechanical description of physical reality be considered complete? Physical Review A, 47:777, 1935.

[3] Matthew B. Elliott. Stabilizer states and local realism. Ph.D. thesis, arXiv:0807.2876v1, 2008.

[4] D. Gottesman. Class of quantum error-correcting codes saturating the quantum hamming bound. Phys. Rev. A **75**, 54:1862, 1996.

[5] D. Gottesman. Stabilizer codes and quantum error correction. Phd Thesis, quant-ph/9705052, 1997.

[6] M. Hammermesh. *Group Theory and its Applications to Physical Problems.* Dover, New York, 1989.

[7] Arnold J. Insel, Lawrence E. Spence, and Stephen H. Friedberg. *Linear Algebra.* Pearson Education, 2003.

[8] J.S. Lomont. *Applications of Finite Groups.* Dover, New York, 2003.

[9] Michael A. Nielson and Isaac L. Chuang. *Quantum Computation and Quantum Information.* Cambridge University Press, 2000.

[10] Robert W. Spekkens. In defense of the epistemic view of quantum states: a toy theory. Phys. Rev. A **75**, 032110, 2007.